

**ELECTRONIC COMMERCE SYSTEMS AND PROCESSES, ESPECIALLY FOR
THE CABLE TELEVISION INDUSTRY**

The present invention relates generally to electronic commerce and more specifically to methods and systems for providing inventory management, video distribution and billing presentment for the cable television industry.

Related Application

This application claims priority to U.S. Patent Application No. 60/122,136 filed February 26, 1999, entitled "Electronic Data Interchange Interconnect Systems and Methods for Broadcast and Cable Television," which is incorporated by reference herein.

Background of the Invention

Processes and systems according to the present invention are useful for electronic commerce in a number of industries and fields. They provide better ways to manage information and workflow, conduct transactions, and distribute product. They are particularly well suited to the cable television industry, which will be used as an example for purposes of discussing the background, summary and disclosure of the invention, subject to the understanding that the invention is not limited to this industry. In the cable television industry, buying, selling, paying for and distributing commercials and advertising are activities that involve many people, considerable coordination effort and talent, and considerable workflow and financial effort. The following is a short glossary for this industry of general importance to the present invention:

- **Advertiser** – any organization or business that has a need to promote themselves, their goods, and/or services: A promotional media consumer such as a car manufacturer, a grocery chain, or a restaurant.
- **National Advertiser** – an advertiser who promotes products in the national marketplace: Typically large manufacturers of consumer goods (soap, toothpaste, preprocessed foods), large restaurant chains, car manufacturers, government agencies, international charitable organizations, and others.
- **Advertising Agency** (agency) – an organization that helps advertisers to promote themselves, their goods, and/or services.
- **National Agency** – an agency that addresses the promotional needs of large national advertisers.
- **Rep Firm** (rep) – an external sales group that represents one or more promotional media providers.
- **National Rep Firm** – a rep firm that promotes local media providers to the national advertising market.
- **National Cable Rep** – a national rep firm that represents local cable providers to the national spot market.
- **Promotional Media Provider** – a group that provides access to some form of promotional media, such as a TV station, a radio station, a broadcast network, a cable network, a newspaper, a cable system, internet/television convergent media, internet media, and others.
- **National Media Provider** – a promotional media provider that serves a national audience: Groups like the major broadcast networks (ABC, NBC,

CBS ...), national cable networks (CNN, ESPN, THE WEATHER CHANNEL...) and convergent media which make at least partial use of the internet are examples of national media providers.

- **Local Media Provider** – a media provider which serves only a local market like a local radio or TV station, a newspaper, cable system or convergent or internet presence.
- **Local Cable Media Provider (LCMP)** – A local promotional media provider whose medium is specifically cable TV. Many times these are simply advertising sales departments organized within the infrastructure of a local cable TV system.

Perhaps the most relevant set of entities for purposes of discussing the background and summary of the present invention includes the Agencies, Reps, and LCMPs. This is because solutions according to the present invention are particularly well suited to smooth their operational bottlenecks through electronic commerce and business computing solutions.

A Conventional Way

Typically advertisers budget a certain amount of money which they are willing to spend towards promotion. Agencies help advertisers use their promotional resources effectively. Agencies assist advertisers by developing campaigns, producing promotional materials, and purchasing promotional media on their behalf. Agencies are compensated for their efforts in media purchase by a taking a commission off the top of the ads that they buy. This cost is not passed on directly

to the advertiser. Rather, the media providers, in this case LCMPs, absorb the cost of the agency's commission. These relationships are similar to those that occur between travelers, travel agents, and airlines. The airlines absorb the cost of travel agents' commissions for buying tickets on behalf of travelers. To a traveler, like an advertiser, the cost is theoretically the same whether the buy is direct or through an agent.

However, unlike buying airline tickets where a traveler typically knows where they want to go and when, advertisers rarely know the specifics pertaining to which networks, times, or media providers may best serve their promotional needs. These decisions are typically left to an agency, which is armed with research data on the effectiveness of different forms of media towards promoting different products. In the case of electronic media, they buy volumes of research data on which programs, networks, and times most effectively reach which demographic segments of the population. They even have large quantities of research data showing which demographic sections of people most often purchase which products.

Advertisers pass on broad-brush information to the agency, such as the products and services they aim to promote, their promotional budget, an image they're trying to portray, etc. The agency transforms this information into a campaign, which normally includes plans to produce commercials, specific promotions, and media spending plans. Media planners, within the agency, determine where to spend the advertiser's promotional money most effectively. Many times the promotion may take place over several different media forms among which may be local, national, and syndicated broadcast TV, radio, newspapers,

coupon distribution, direct mail, cable TV networks, local cable TV, and many other forms of promotional media too numerous to mention. Wise media providers and their reps try to get involved in the media planning process to insure that they are considered in the buying process. Accordingly, media reps spend much of their time promoting the media they represent to planners. Such material might include presentations of demographic, audience, product, and brand research, along with various buying proposals. If the reps are fortunate, or they succeed in convincing the planner, they are ultimately included in the media buy. Once the advertiser approves the media plan, media buyers go ahead and place orders with media providers.

When a national agency places orders with local media providers, it is usually due to a constraining set of circumstances. This is because it is much easier to place ads with national media providers, which yield larger audiences, than it is to deal with numerous local providers. After developing the advertiser's media plan for the year, the agencies' buyers purchase media from the national providers in large blocks during the up-front sales season, which precedes the advertising year.

When a national agency places local media buys it is called "National Spot Advertising". The "Spot" portion of the name is due to the local media being purchased in smaller blocks to: shore up larger national campaigns, target smaller geographical regions, or accomplish marketing objectives that have changed since the beginning of the TV season, when major up-front spending commitments are usually made. Solutions according to the present invention are particularly useful for facilitating national spot advertising on local cable television media.

Reps have the task of promoting various media providers in a favorable light to agencies in order get more national spot money flowing in their direction. Most reps act as external sales forces on the behalf of media providers, in this case specifically local cable media providers (LCMPs). Most national spot orders on local cable pass through both an agency and a rep. This is largely due to the complexity of placing orders with local cable media providers (LCMPs). Usually an LCMP will only deal with a single rep firm, which has an exclusive agreement to represent them to the national spot market.

At any given moment current local television viewership is normally divided (very) roughly at a 60/40 split between local broadcast and local cable providers, although that will change. Local broadcast viewership is typically shared between a few major network affiliates and a handful of local independent TV stations. However, all of their signals will generally cover the entire metro area.

Metro cable viewership, on the other hand, is normally shared between several different cable providers with their signals being constrained to only their cable runs. Cable runs are confined to strict geographical boundaries dependent on their franchise area(s). In contrast, broadcast signals travel over the whole metro region through the air. Additionally, a single cable TV provider may have several signal distribution points within their franchise areas. Then at each of their signal distribution points they will carry advertising on numerous networks, at least on the ones permitting advertising (CNN, ESPN, THE WEATHER CHANNEL, etc.).

It is therefore much easier for an agency to place orders with a single TV station and get full market coverage than in the same market to be forced to deal

with numerous LCMPs and their numerous networks. For this reason, reps play a large role in national spot on local cable.

Reps are compensated in much the same way that an agency is compensated for its media buying services. They take a commission off of the orders that they place and bill back to the agency. So by the time an LCMP accepts an order from a rep, for national spot, they are committing to absorb the cost of paying at least two commissions, one to the rep, and one to the agency.

The agency places one order with the rep and the rep, in turn, assumes the responsibility of breaking one order into many contracts and distributing them to pertinent LCMPs. Because there may be many individual cable providers in any given metro area (national spot buys are usually concentrated in the larger metro areas) an agency order can turn into as many as 20 contracts. Reps typically distribute contracts to the providers via fax.

Once a media provider receives a contract, it is their responsibility to confirm back to the rep as to whether they accept it or not. They normally must accept a contract by signing its signature page and faxing back to the rep. Other times they pursue further negotiation to reach a more acceptable arrangement. In any case, once both the rep and the provider agree confirmation is made via signature and fax.

Once a provider accepts a contract, they key it into their traffic and billing (T&B) system. They use their T&B systems to schedule the ads (spots) they are to air. T&B systems are usually capable of communicating with a provider's playback system, which eventually plays the commercials on the air (cable).

Advertisements or commercials are often referred to as "Copy." Once the orders are placed and the commercials are produced, the agency then has to distribute them to the media providers prior to their scheduled airtime. Instructions as to where the commercials are to be aired, at which times, dates, weekdays, etc., is of primary concern to. These so-called "Copy Instructions" have to be passed on to the LCMPs so that they properly schedule the commercials to air. At present, paper based copy instructions typically accompany commercial tapes, which are transported by commercial overnight shippers and carriers, all over the country. One of the major potentials for efficiency according to the present invention is to shift this Copy or content delivery from shippers and carriers to online distribution.

LCMPs' T&B systems make records (logs) of the commercials they air for each contract through the end of the billing cycle, usually a broadcast month. They then have the task of reconciling the spots that aired versus the spots that are ordered. This can be a time consuming and error prone process for the provider. Once they are through contract reconciliation, they produce invoices. Reconciliation is a huge subject; suffice to say here that reconciliation is necessary because all of the spots don't air exactly as wished and effort is usually required to make the advertiser whole, either by discounting the bill or airing additional content.

At some point the provider gathers all of the national spot invoices and forwards them to the rep, usually via a mail carrier. Depending upon the attitude of the provider, a few of them attempt to process their national spot invoices prior to processing their much larger job of invoices for their local clientele. Others produce their local invoices first due to the fact they can expect payment sooner from their

local customers than they can from their national spot customers. In either case, they eventually produce all of the national spot invoices and forward them to their rep.

The reps organize the invoices they receive into batches and produce a billing summary for each batch. Each batch is comprised of all the provider invoices pertaining to a single agency order. Of course all of the providers don't forward their national spot invoices on the same date. Given that an invoice batch must be complete before the rep can forward it to the agency, the rep is often forced to wait for straggling invoices that can hold up an entire batch. This introduces some latency into the process. However, the largest portion of the invoices will arrive within 30 days of the end of the billing cycle permitting the rep to forward most of the invoice batches and billing summaries on to the appropriate agencies.

Once an agency receives a batch of invoices from the rep firm, they sit down and key all of the invoice data into their data system. They do this so that their system can then reconcile the spots they ordered against the spots that are billed to them. This can be a very time consuming process due to the size of some of the invoices and there can be many invoices, from different cable providers, issued against any single order. By the time an agency is through most of their reconciliation processes, 60 days will normally pass since the end of the provider's billing cycle for the invoices.

By the time an order is placed, aired, and billed it passes through many human hands and many disparate processes. The invoices tend to have a good number of errors by the time they return to the agency; such as:

- Spots are missed because of playback complexities
- Spots may play on the wrong networks
- The wrong copy is assigned to spots
- Copy does not arrive before an order starts
- 5 • Orders are improperly translated by providers

The agency buyers are typically left having to work out a way to compensate for the lost spots or simply forget about them. Agencies are highly motivated to recover lost spots due to the fact that they are paid on commission. Although they wish to help the advertiser make the most of their promotional dollar, they also want to spend as many of the advertiser's dollars that they can, in order to make more commissions.

To recover lost revenue, they contact the rep to work out a makegood plan. Makegoods are spots that are run to compensate for others that were missed. Once the makegood plan is worked out, the order cycle continues where the rep places new makegood orders with the providers. The providers run the spots and eventually bill them back to the rep, who bills them back to the agency. (Makegood invoices are for zero dollars since they're compensating for previously missed spots.).

Because of the fact that two months normally pass by the time an agency finds out, in reconciliation, that the invoices fall short of the orders, they often lose the opportunity to compensate for missed spots. Most campaigns will end prior to the agency discovering a billing shortfall. So makegood opportunities are slim in national spot cable while incidence of errors resulting in shortfall is high.

When the agency is finally satisfied that they have captured all value that they can on an order, they will compile a bill to present to their advertiser. Once the advertiser receives the invoice from the agency, they remit payment. The agencies skim their commission off of the top of the advertiser's payment and forward the remainder to the rep. The rep also skims their commission off the top and distributes the remainder amongst the providers.

Alltold, in national spot cable, between the time a provider airs spots for an order and the time they get paid from the rep, 6 months will normally pass.

A Better Way

A central theme to systems and processes according to the present invention is to automate and facilitate more efficient workflow, transaction, distribution, and information management processes among the above mentioned entities, preferably using a common document model that allows everyone at every stage in the trading process to avoid replication of effort, draw on an easily accessible (but secure), readily translatable, common data source, and otherwise deal with each other more efficiently and effectively. In some cases, such as an Order Confirmation, such systems and processes can generate electronic documents where no paper document, and perhaps only a verbal dialog, conventionally exists. This is due to the fact that systems and processes according to the present invention can leverage computing power and connectivity to represent trading partners. Use of a common document model further leverages the power of the computer and the global information infrastructure, since it allows everyone's computer to interact using a lingua franca such as for example common document models implemented to

transact in XML or successors, rather than conventional EDI techniques which rely on a host of proprietary, custom implemented platforms and software, each of which must be programmed to talk to (and listen to) all others. Systems and processes according to the present invention are in any event able to accommodate and format electronic documents in whatever format a client will be prepared to receive them and perform necessary value translations as well. For instance, in situations where a conventional EDI ANSI X12 document has been defined and clients are prepared or prefer to trade using X12, then systems and processes according to the present invention can do it. But because of the nature of metatags and the common document model, systems and processes according to the present invention can, unlike conventional EDI techniques, efficiently translate to or from any application specific format a client is prepared to transmit or receive readily, easily and efficiently.

In the case of a current national spot on local cable for instance, no ANSI X12 standards exist presently and only two electronic document exchange formats even approach standardization, and even they are rarely used presently in local cable. Systems and processes according to the present invention can create neo-standards by enabling the regular exchange of electronic documents. Examples of documents in a simple media transaction model which are readily accommodated by systems and processes according to the present invention are:

Avail Request

Potentially generated by media planners and sent to media providers, in this case the rep(s), to gather information about making potential buys helping them formulate their media plan. The rep would normally respond by returning an avail document.

5 Avail

A document, originating with the rep, which proposes an order to an agency. An agency may respond to an avail with an order or not even respond at all. An agency may originate an order to a rep without ever issuing an avail.

Proposal

10 When the rep is negotiating with providers, they may originate this document which proposes to buy spots at the specified terms. A proposal is very similar in form to an order but does not constitute an agreement to buy spots. A proposal does not always result in a contract and contracts may be sent to providers without preceding proposals.

15 Order

Denotes the commencement of trade between agency and rep partners in national spot buying. It will originate at a buying agency, which will transmit orders to a rep firm. An order will describe the desired purchase of airtime (spots) at targeted cable providers. It will include market (DMA), acceptable networks, date ranges, time
20 ranges, weekdays, lengths, spot rates, etc.

Contract

These documents will originate with reps and will comprise their attempts to agree with providers on airing spots from agency orders. They will contain largely the same information as the agencies' orders but will be produced uniquely for each provider.

- 5 A rep may sequentially issue multiple contracts from the same order to a provider until the provider finally accepts via confirmation.

Confirmation (Decline)

Comprises a provider's acceptance or rejection of a proposal, contract, or contract change to a rep. When a contract, proposal, or change is accepted, the document reflects all of the contract information that the provider accepts. When declining, the confirmation will contain only those elements that the provider disputes and possibly their proposed changes. Proposals, contracts, and contract changes must always be wholly accepted or declined.

Agency Contract

- 15 Originating with the rep, these will signal success or failure to the agencies in placing all of the necessary provider contracts to fulfill their orders. They will reflect each of the ordered elements and to what degree they were successfully contracted.

Modification

- 20 A document, from an agency, to a rep, that changes their original order. These reference only elements of an order they wish to change (or cancel) and how they would like to change them.

Contract Change

Will be issued by a rep to a provider in response to a change order from an agency.

These will contain largely the same information as a change order but geared towards each provider.

5 Copy Instructions

Will originate with an agency and list, in detail, which specific commercials should be aired in conjunction with their orders, and on which networks, dates, times, etc.

These will transfer directly from the agency to the provider without passing through the rep.

10 Affidavit

A bill, issued by a provider and destined to their rep, which details all of the spots they have aired in relation to a single contract and billing cycle. In addition it will reflect information such as networks, copy, airdates, air times, commercial scripts, lengths, etc.

15 Invoice

A bill, issued by the rep to the agency, for the spots aired on behalf of their order for a single billing cycle. It will contain largely the same information as an affidavit, excepting an accompanying fulfillment summary.

Agency Remittance

20 A document issued by an agency to a rep indicating how much money has been deposited in their account to fulfill payment of an invoice.

Rep Remittance

Will be issued by a rep to providers indicating how much money is deposited in their accounts, relating to affidavits.

Current Operational Deficiencies

5 It is clear from the discussion above that there is significant opportunity for efficiencies in the ordering, distribution and billing processes of national spot cable.

Among such opportunities are:

- Orders are keyed in manually at the rep before passing on the providers yielding an opportunity for human error.
- 10 • Contracts are sent to providers via Fax.
- Contracts are retyped into the providers' T&B systems introducing yet another opportunity for error and no confirmations are sent back to the rep indicating how the providers key in orders. Thus there is no opportunity to catch keying errors by the providers.
- 15 • There is so much latency in the process of getting orders all the way down to the providers that order changes are nearly impossible.
- Copy instructions are sent on paper usually with the commercial tapes but not always. There is not a 100 percent standardized procedure.
- There is no practical way for an agency or rep to verify that a provider has
20 received the appropriate copy for an order. When someone at the provider finally notices they're missing copy they have to call back the agency and have the copy forwarded, usually at least an overnight process.
- Provider reconciliation procedures are not always reliable.

- Provider invoices are on paper and are often submitted late holding up the procedure.
- Provider invoices come in many varying forms. They are not standardized requiring the reps and agencies to decipher the varying formats.
- 5 • Invoices are manually reviewed by the rep because they do not have the resources to do a detailed reconciliation. There is also some doubt as to the value of this potential procedure because the agencies perform a detailed reconciliation anyway.
- 10 • Reps are required to wait for complete invoice batches before forwarding them to the agencies.
- Invoices are manually keyed at the agency, which introduces even more errors.
- Reconciliation is so slow that makegoods are often infeasible.
- 15 • Payments pass through several hands each taking time to distribute funds manually.

Summary of the Invention

20 Systems and processes according to the present function in the cable television industry to automate and promote efficiency in advertising activities, including (1) advertising strategy planning and implementation; (2) advertising sales and placement; (3) advertising content creation and distribution; and (4) invoicing, billing and payment for advertising. These systems and processes are therefore

well-suited for the cable television industry as it exists contemporaneous with the date of this document. They will be just as well or even better suited for advertising and other content activities in electronic media of the future, including the television, internet on home server / network, computer or set top box, convergent media on

5 home server / network, computer or set top box, wireless media such as hand phones, personal digital assistants, pagers, simple messaging systems, and future successors to any of these. In a broader sense, systems and processes according to the present invention can be adapted to be useful in any field that contains disparate parties who must transact efficiently with each other, perhaps most

10 relevantly in a context where product is also distributed.

For a contemporaneous architecture as one context for disclosing a particular embodiment in which such systems and processes could manifest themselves currently, consider the cable television industry circa change of the millennium. Systems and processes according to the present invention can be implemented in

15 any or all phases of advertising in this media, including, as recited above: (1) advertising strategy planning and implementation activities, involving advertisers and ad agencies and requiring communications and interaction between them; (2) advertising sales and placement activities, including contract negotiations to buy ad inventory and other involvement by any or all of the group of advertisers and / or ad

20 agencies, a broker such as NCC, and media owners or providers such as MSO's or cable companies; (3) advertising content creation and distribution activities involving any or all of production entities, distribution entities, brokers and media owners or

operators; and (4) invoicing, billing and payment for advertising activities, involving any or all of the foregoing entities.

Systems and processes according to the present invention can use a common document model so that information combined with metainformation, may be used intelligently, efficiently and effectively for various purposes, by various entities running a variety of proprietary or nonproprietary systems, to accomplish communications, negotiation, content distribution, and EBPP among other tasks requisite to carrying out the present invention. The following example continues the discussion relative to the media transactions discussed in the Background section, as merely one example of systems and processes according to the present invention.

Reps can prepare avails on their own in-house computer systems if desired, or these can be prepared, stored and processed online (as can any other transaction or task discussed in this document) on a master platform (which can take the form of a network of servers and / or other computers, memory devices and other functionality, a single server, or as otherwise desired, in one or many geographical locations) according to the common document model and forwarded to agencies over the global information infrastructure as desired. The master platform can be connected by any desired transport infrastructure or capacity, including internet, private network, or otherwise, to client platforms at the agency, the rep, media providers, and other entities participating in the systems. Such client platforms can take the form of networks, single computers, or any other functionality; the browser or its successor can present the GUI or any other interface that is desired ("GUI").

The master platform can correspond to the function of current NCC roles and responsibilities, if desired. Any authorized party at the rep can monitor status of the avail via client GUI. They may also request to re-send, review, and print any previously transmitted document.

5 Once in the hands of the agency, the authorized target may review, edit, print, and export the avail to their stewardship system or otherwise operate on the avail via their GUI connected to the master platform or the agency GUI. Edits will be permitted on avails to allow the agency flexibility prior to importing them into their stewardship system and processing them as orders. If they are satisfied with the
10 avail, they may accept it and forward it back to the rep as an order directly from the master platform without having to travel back and forth between their stewardship system. They may also directly reject an avail and automatically forward back to the rep if desired. The agency client applications can also read any orders exported by an agency stewardship system and forward them to the rep.

15 During the process of negotiating an order with an agency, the rep may wish to get tentative agreement from the local providers for the purchase of spots. Once proposals have passed all of the rep's in-house requirements for transmission, they can automatically be forwarded to appropriate providers. The forwarding process can automatically map rep values for networks, zones, and other crucial data
20 elements to values expected by the providers.

Providers may review and print the proposals with using their client systems according to the present invention. They can agree with the terms and accept proposals unchanged, disagree and decline proposals outright, or edit proposals with

desired changes. In each case they can receive an online confirmation with the providers' decisions and potentially their changes.

When declining confirmations arrive at the rep they can be readily and automatically translated to a format compatible with their in-house computer system because of having been stored and processed in accordance with the common document model. There they can issue new proposals incorporating changes and continue the proposal/confirmation negotiation process with the providers.

Once an order is received by the rep client from an agency it can be translated automatically and converted to a format that can be directly imported by the rep's in-house computer system readily and easily because of having been stored and processed in accordance with the common document model. Within their in-house system the rep will generate contracts for the providers based on the orders they receive. These can then be forwarded to the appropriate provider.

Contracts, like proposals, without accepting or rejecting them, may be reviewed and printed by the providers. However, if they wish to accept them, they may not (usually) be edited. If they choose to decline, they may edit the contract with hoped for changes. However, the provider will have to supply their customer and order numbers so that agency estimate and customer numbers can be correlated to provider order and customer numbers for future translations. And again like proposals, confirmations are automatically forwarded to the rep upon acceptance or decline. When the provider accepts a contract they may choose to export it in a format compatible with their traffic and billing system to avoid the painful and error-

prone re-keying of orders. Again, this is possible through use and leveraging of the common document model in the present invention.

When the rep finally receives in-house all of the accepted confirmations relating to an order, their in-house system will generate an agency contract which
5 can be forwarded to the agency, either from the rep's system or the master platform. Agency contracts may be reviewed and printed at the agency as well as exported to their stewardship systems for order processing.

Agencies can prepare copy instructions for their orders on their client system or the master platform for use in accordance with the common document model.
10 Once they are satisfied with the copy instructions they may opt to forward them to the providers. The master platform can determine which providers require the copy instructions, based on the contracts generated by the rep, and forward them to their many destinations from a single agency command.

Providers, once they receive the copy instructions, may translate them or,
15 where supported by T&B vendors, export the copy instructions in a compatible format.

When an agency wishes to change an order they have already issued, they can generate a modification either through their stewardship system or directly into their agency client application or the master platform. The system can automatically
20 transport the modification to the rep. The rep, with their in-house system, should receive the modification and generate contract changes destined for each provider. These contract changes can be forwarded to each provider where they can treat it in much the same manner as they will contracts. They can either accept or reject it and

move it into their T&B systems. Again, because of the common document model, it is straightforward and efficient to translate easily into format and protocol usable by conventional and proprietary T&B systems if desired. Confirmations for the contract changes can be automatically forwarded back to the rep. The rep can, in-turn,
5 forward a new agency contract, which includes their modifications, to the agency once they have received all of the necessary confirmations.

At the end of each billing cycle (normally a broadcast month) and at the end of each contract's flight, after the providers have aired the contracted spots, they produce affidavits, bills which list they aired spots, for all of their clients, both local
10 and national spot. For the national spot clients they will produce electronic affidavits. The provider client according to systems and processes of the present invention allow the providers to review, edit, and print affidavits online or at the provider prior to transmission to the rep. Within the transmission, systems and processes according to the present invention can perform value translations on selected fields
15 of the documents including customer number, order number, network, system, etceteras, to values understood by the rep and agency systems. Once affidavits are transmitted, provisions can be made for the providers to review their payment status.

After affidavits are ready for forwarding, they are forwarded to the rep. The rep, with their in-house computer system, can perform reconciliation of the affidavits
20 against their contracts and add a fulfillment summary section in order to generate an invoice. This section summarizes the provider's performance against their contract. This new invoice is forwarded to the agency as soon as it is generated without waiting for other providers' affidavits from the same order. This will allow the agency

to proceed immediately with their reconciliation without having to wait, as is the case today. Reconciliation can occur online and / or automatically on the master platform, where all avail information, contract information and affidavit information is already stored according to the common document model according to the common document model.

As invoices arrive at the agencies' clients, they will be able to review and print any or all invoices received. After exporting them to their stewardship systems, which again can occur easily because the information has been stored and processed according to the common document model, they may perform their reconciliation processes to verify that all of the ordered spots are aired. When invoices fall short of expectation, they can submit makegoods, in the form of modifications, to start the process over again.

When the agencies are satisfied that they are ready to collect, they bill their advertisers. When the advertisers remit payment, the agencies will remit payment, less their commission to the reps and will send remittance of the same through EBPP or other payment systems which are part of systems and processes according to the present invention. For instance, when the rep client receives the remittance, it can be automatically translated to a format suitable to the rep's computer system for import. The rep's system should then create remittance documents to automatically forward to the provider clients. And finally, at this point, the provider client can automatically reconcile the payment against the originally transmitted affidavit so that when a provider wishes to review payment status, it will show that remittance has been received. Once again, the common document model according to which all

relevant information can be stored, makes this task efficient and straightforward to occur in an automated way. The client can also export the remittance documents as payment transactions to the provider's T&B system.

As discussed below, systems and processes according to the present invention can also include the ability to distribute advertising content or copy to the providers electronically.

Thus, use of a common document model allows each of these steps to occur in an automated way, either locally or on the master platform, online or offline. Because systems and processes according to the present invention capture information at every step and can store it according to the common document model, reconciliation and comparison of data allows contract negotiation, editing, reconciliation, bill preparation, and other activities to occur easily and efficiently, and for tasks or portions of them to occur either on the master platform or on clients to which the master platform can easily communicate via XML or other metatagged data and therefore readily transformable date. A particular beauty of systems and processes of the present invention is that they make it possible for various platforms, legacy T&B systems, and other proprietary platforms to talk to one another in a lingua franca such as XML or successor and all leverage the power of a common document model database, whether online or offline.

The following chart is a brief summary of the document flow discussed above, any or all of which can occur on systems and processes according to the present invention.

Proposed Document Flow Summary

Document	Originator	Destination
Avail Request	Agency	Rep
Avail	Rep	Agency
Proposal	Rep	Provider
Order	Agency	Rep
Contract	Rep	Provider
Confirmation (Decline)	Provider	Rep
Agency Contract	Rep	Agency
Modification	Agency	Rep
Contract Change	Rep	Provider
Copy Instructions	Agency	Provider
Affidavit	Provider	Rep
Invoice	Rep	Agency
Agency Remittance	Agency	Rep
Rep Remittance	Rep	Provider

Workflow Requirements

- 5 The following is a summary of some major workflow requirements in the example discussed above which can be accommodated and improved using systems and processes according to the present invention.

Agency Client Requirements

Avail Requests (Outbound)

- Import avail requests from stewardship system
- Review, edit, and print avail requests
- 5 • Transmit avail requests to multiple destinations

Avails (Inbound)

- Receive avails in queue
- Review, edit, and print avails
- Accept avails and auto-generate orders
- 10 • Export avails to stewardship system

Orders (Outbound)

- Translate exported orders from stewardship
- Review and print orders
- Queue orders for transmission

Agency Contracts (Inbound)

- Receive contracts in queue
- Review and print contracts
- Export contracts to stewardship system

Copy Instructions (Outbound to Providers)

- Review, edit, and print copy instructions
- Queue copy instructions for transmission
- Auto-determine recipient provider list

5 **Modifications (Outbound)**

- Review, edit, and print contract modifications
- Queue modifications for transmission

Invoices (Inbound)

- Review and print Invoices
- Export invoices to stewardship system

Rep Client Requirements**Avails (Outbound to Agencies)**

- Periodically scan for outbound avail documents
- Auto-transmit avails to agencies

15 **Proposals (Outbound to Providers)**

- Periodic scan for outbound proposals
- Auto-transmit proposals to providers

Orders (Inbound from Agencies)

- Receive orders in queue

- Auto-translate queued orders to in-house form

Contracts (Outbound to Providers)

- Periodic scan for contracts
- Auto-transmit contracts to providers

5 **Modifications (Inbound from Agencies)**

- Receive modifications in queue
- Auto-translate modifications to in-house form

Contract Changes (Outbound to Providers)

- Periodic scan for contract changes
- 10 • Auto-transmit contract changes to providers

Confirmations (Inbound from Provider)

- Receive confirmations in queue
- Auto-perform key value mappings
- Auto-translate confirmations to in-house form

15 **Agency Contracts (Outbound to Agencies)**

- Periodic scan for agency contracts
- Auto-transmit agency contracts

Affidavits (Inbound from Providers)

- Receive affidavits in queue

- Auto-perform key value mappings
- Auto-translate affidavits to in-house form

Invoices (Outbound to Agencies)

- Periodic scan for invoices
- 5 • Auto-transmit queued invoices

Provider Client Requirements

Proposals (Inbound)

- Receive inbound proposals in queue
- Perform necessary value mappings on proposals
- 10 • Review and print proposals
- Accept or decline proposals
- Edit and add comments to declined proposals
- Require providers to supply key value mappings on accepted proposals

Contracts (Inbound)

- 15 • Receive inbound contracts
- Perform necessary value mappings on contracts
- Review and print contracts
- Accept or decline contracts
- Edit and add comments to declined contracts
- 20 • Export contracts to T&B system

- Require providers to supply key value mappings on accepted contracts

Contract Changes (Inbound)

- Receive inbound contract changes
- Perform necessary value mappings on contract changes
- 5 • Review and print contract changes
- Accept or decline contract changes
- Edit and add comments to declined contract changes
- Export contract changes to T&B system
- Require providers to supply key value mappings on accepted contract
10 changes

Confirmations (Outbound)

- Auto-generate confirmations on proposals, contracts, and contract changes
- Auto-queue confirmations for transmission to rep
- Print confirmations
- 15 • Auto-transmit queued confirmations

Copy Instructions (Inbound)

- Receive transmitted copy instructions in queue
- Review/Edit/Print Copy Instructions
- Export Copy Instructions for T&B Input

Affidavits (Outbound)

- Capture affidavits from T&B export into a batch
- Review, edit and print single or sorted affidavits from a batch
- Queue affidavits for Transmission
- Query payment status on transmitted affidavits

Remittance

- Review and print received remittances
- Auto-update affidavit payment status on received remittances

General Client Requirements

- Perform key value mapping on appropriate data elements
- Archive all inbound and outbound documents
- Ability to review and print all archived documents singly or from groups
- Retransmit any archived outbound documents
- Retranslate any inbound archived documents
- Auto-generate and transmit acknowledgements of inbound documents
- Perform acknowledgement audits on outbound archived documents
- Auto-transmission of queued outbound documents
- Auto-retrieval of inbound documents
- Auto-retransmission of unacknowledged documents

Administrative Requirements

- Capture statistical data for client billing

- Maintain archives of all In and Outbound documents
- Ability to retransmit any document from any source
- Perform audits of documents
- Client billing
- 5 • Return Ax to all document originators

Systems and processes according to the present invention are particularly well suited to the advertising field and cable television advertising in particular. Such systems and processes are not limited to those fields, though. They are suitable for any market or field of endeavor characterized by some or all of the following features and factors, among others: (1) the inventory to be sold is highly perishable; (2) the inventory to be sold is very sensitive to the time at which it occurs and in which context; (3) buying and selling the inventory involves significant back and forth correspondence; (4) the inventory for a single effort usually must be negotiated for and bought from multiple entities; (5) content or product must be delivered to each of the multiple entities in order leverage the value of the inventory bought and sold; (6) there are usually discrepancies between what inventory is bought and what is actually delivered; (7) these discrepancies affect the amount to be paid and require additional negotiations; (8) there is efficiency to be gained by centralizing communications and information flow.

Brief Description of the Drawings

Fig. 1 is a functional block diagram of an exemplary electronic commerce system for use in the cable television industry.

Fig. 2 is a functional block diagram of a first embodiment of an information management system according to the present invention.

5 Fig. 3 is a functional block diagram of a first embodiment of a system for distributing content according to the present invention.

10 Fig. 4 is a functional block diagram of an embodiment of an electronic commerce system according to the present invention that includes an information management system and a content distribution system.

Fig. 5 is a functional block diagram of an embodiment of part of the back end of an electronic commerce system according to the present invention.

15 Fig. 6 is a functional block diagram of an embodiment of part of the front end of an electronic commerce system according to the present invention.

20 Fig. 7 is a functional block diagram of an embodiment of part of the front end of an electronic commerce system according to the present invention, focusing on payment issues.

Detailed Description

Conventional Solutions

Systems and processes according to the present invention, broadly speaking,
5 facilitate electronic exchange between large trading partners such as rep firms and
large advertising agencies as well as some relatively small trading partners such as
cable systems and television stations. Conventionally, some solutions have already
implemented some non-standard forms of electronic data interchange ("EDI") on a
very limited basis while others have not. Progress was slower than anticipated
10 because of all the problems associated with EDI techniques including proprietary
technology, communication and standardization issues and other well known issues.
One conventional approach to this problem was to undertake the burdensome task
of performing translations between unlike systems that did not and may never speak
the same language. This approach included an electronic mailbox system, a
15 universal document translator, a value mapper, an auto-targeter, an Internet-based
management tool, and a high volume interface. Very briefly, universal translation
involves translating documents entering a system from an application form into a
common form and translating documents leaving the system from the common form
into another application form. Value mapping involves adjustment of code set
20 incompatibilities between information systems. For example, no two systems call
"Headline News" the same thing; some use "HDLN", others "CNNH", and even
"HLN." The issue is that if one application system receives a document that contains
a different abbreviation for Headline News than it expects, bad things will happen.
Value mapping looks for values that it knows have the potential to be mapped and

keeps a list of what each trading partner calls the same thing. Mapped values within inbound documents are changed to contain common values. When a document is on its way out its mapped values are changed to agree with the destination system's expectations. Auto targeting involves examining documents as they move into the system to determine based on the document contents who the intended target (destination) should be. This technique aims to reduce or eliminate manual interpretation. The conventional management tool is a browser-based Java applet that allows users to see, edit, print, and control all of their documents as if they were using an e-mail system. The high volume interface allows large trading partners (such as rep-firms) who have MIS staffs to directly connect their applications to the systems. This obviates the need for several components that accompany a standard EDI implementation. It also accommodates the special needs of high-volume trading partners that are unlikely to manage their electronic trade one document at a time. The first generation trading system enabled facilitation of millions of dollars in media trade electronically every month. It has put EDI at the fingertips of trading partners to whom it would normally be inaccessible.

A functional block diagram showing the first generation system is FIG. 1. Its main strengths are:

- Requires no MIS staff.
- Inexpensive.
- Centrally Maintained.
- Scales down to the smallest trading partners yet supports high transaction volume.

Systems and Processes According to the Present Invention

Even though the conventional solutions solved numerous problems with electronic trading in the media industry, they leave room for improvement. Systems and processes according to the present invention can accomplish the following:

- Extend electronic trade to those media trading partners who lack EDI interfaces or even automation systems.
- Remove the constraints imposed on electronic trade by application system vendors.
- Facilitate a complete media transaction set including orders, contracts, confirmations, changes, makegoods, and invoices.
- Help media trading partners eliminate filing cabinets and have immediate and searchable access to their document archives.
- Eliminate "missed steps" and "dropped balls" in trading by giving every person who trades in media a 'To-Do' list that highlights documents (trading opportunities) that require their attention.
- Enable people in the media industry to trade electronically in the way that comes most naturally as they conduct their ordinary business everyday.
- Simplify EDI through the use of direct application interfaces.

This solution is a much richer implementation of electronic trade than EDI. It may be referred to as electronic trade management (ETM), which is more inclusive than EDI. A very high level functional block diagram which illustrates, in some respects, ETM, is shown as Fig. 2.

Using the solution shown in Fig. 2, a common document model or other system for storing and processing data with associated metadata allows for storage and processing of documents and information for conveyance according to a "lingua franca" such as XML or successor-based implementation, thus for ready and straightforward interpretation into any format desired to accommodate any third party proprietary or nonproprietary platform, standard, or interface. Such information is, for instance readily available on browser based implementations, rather than requiring complex, expensive and proprietary conventional EDI platforms and processes.

In the implementation shown in Fig. 2, which is only one way systems and processes according to the present invention can currently manifest themselves, a business to business tracking server implemented in the form of a business to business collaboration toolkit ("BOBCAT") interfaces with a variety of third party platforms as well as various support databases and support functionality, including XML or other common document model databases. The server may be accessed using GUI interfaces such as browsers, so that access can be ubiquitous, straightforward, quick and secure. The BOBCAT server may in some respects be thought of as a workflow server. Attributes of workflow in media content sales and distribution include: (1) Workflow formally models the business process; (2) Depends on work being broken down into several ordered steps; (3) Work is completed by the collaboration of many individuals with specific skills (or authorities); and (4) The collaboration often involves several businesses including media outlets and agencies. The BOBCAT server implements systems and application that carry

out all necessary steps in the workflow process, and that allow access to documents and data at every stage of a particular workflow process in order, for example, to negotiate and buy advertising or to pay for it.

One advantage of solutions according to the present invention is that they can track documents and every revision of every document exchanged in their on-line archives. Each document may be stored in an XML-based form that enables very efficient search engines ready access to all documents. Users may query the archives for any document they have ever traded and view it, print it, resend it or otherwise use or operate on it. This robust functionality replaces filing cabinets and saves time in searching for documents reflecting past transactions or a particular transaction.

Another advantage is that the common document model allows the BOBCAT platform to communicate easily and efficiently with client platforms and legacy and proprietary T&B and other systems.

Another advantage of solutions according to the present invention is push: operators logged into the system can have a 'To-Do' list which contains every document or pending transaction requiring their attention. These documents remain in their 'To-Do' list until they take an appropriate action. As soon as they act upon the document(s) they are removed from their 'To-Do' list, likely landing in someone else's. Current trading requires that people keep track of all of their trading activities. Keeping faxes, e-mails, voice-mails, phone-calls, paper forms, napkins, and sticky-notes organized requires planning and organization. In typical trading roles, things

fall through cracks, and are lost or postponed. Common document model solutions according to the present invention potentially eliminate these unpleasant issues.

Yet another advantage of solutions according to the present invention is that they allow people to work naturally. A workflow engine can allow for configuration of the system to match each operator's needs and workflow. For example, a trading partner can indicate all of the active trading roles that are currently active, such as buyer, account executive, sales assistant, sales manager, traffic manager, etc. The workflow engine is programmed to know what actions each role may perform in any trading transaction. Then, the workflow engine can be programmed to know how documents move between the different trading roles in an organization. For example, when a sales assistant submits a contract, it moves to the responsible account executive who must approve it. When the account executive approves it, a confirmation is forwarded to the buyer. Thus, solutions according to the present invention allow people to trade in the natural way that they work except that electronic documents flow between trading partners (even within the same office) rather than voice mails, paper forms, pink slips, post it notes and faxes.

Other advantages of systems and processes according to the present invention are that they:

- Require little or no adaptation of current business processes, automation systems, and trading partner relationships. Facilitating direct document exchange between applications is the best way to accomplish this feat. Documents are taken from and delivered to ad agencies, reps, media providers and other customers in formats that are native to their own automation systems.

In cases where there is little or no automation, lightweight applications can be provided. These will allow potential users to create, edit, and manage electronic documents with very little investment of time or capital.

- Does not require customers to change their trading relationships significantly, thereby promoting use.
- Non-intrusive, requiring little or no system maintenance, requiring little or no support staff, thus further reducing customer expense and difficulties.
- Requires merely the capability to run browser software will maximize its availability by minimizing entry requirements.
- Intuitive and simple to use, GUI browser based, or applet enabled, much easier than their existing trading schemes.
- Easy enrollment over the Internet, without requiring any special installation or training to immediately engage services to transport simple documents.
- Responds rapidly to customer requests or input. (Barring bandwidth issues for Internet based customers)
- Performs well for light clients over the Internet.
- Secure document trading.
- Extensible support for new document types and trading partners with relative ease.
- The system accommodates changes in documents and procedures.

The term "client" is used herein in the "client/server" context, to refer to a software application, rather than a customer. (In this specific case our client software happens to be used by our customers, or clients, but this is irrelevant to the

discussion.). In this context, the client software is a client, or satellite, of the server software. Client software will allow users to create, edit, and perform all kinds of functions relating to electronic documents, but the documents will reside on the operations center server(s).

- 5 Given the maintenance and support concerns over the far-flung nature of the entire enterprise, coupled with the aim to make applications as non-intrusive as possible to customers, client applications are kept as lightweight, or as "thin", as possible. To maintain software in a constantly functional state on literally hundreds of computers simultaneously would be a daunting task using conventional methods.
- 10 Accordingly, systems and processes according to the present invention build this infrastructure adhering to a strict division of labor between the client and server. The client software is primarily concerned with document presentation, capture, and controlling management operations (not performing management operations). This implies that much of the functional weight of the system may be borne centrally by
- 15 servers, which will reside at the operations center on the master platform.

The only requisite support software for client applications, will be applet including if desired Java or subsequent-capable web browser. Client programs that the user will see will be uploaded to their web browser on demand. To reduce network traffic the server can reload only applications that have been previously

20 uploaded, if they are changed. When a client program is changed, it can be loaded on the server and then anyone who tries to use it will receive the update automatically.

Although users will edit and invoke document operations through their client software, the documents themselves will reside on master platform server(s), where the most significant operations against them will be performed. When a user views, prints, or edits a document, it will be uploaded at that time to their web browser.

5 Commands issued to save or transmit a document are executed at the server. In other words, the client applications permit a user to view or control operations on a document but the real work takes place on the servers.

Of course users of client software will be able to create new documents, like contracts or orders, on-screen, should they so desire. However, another solution for
10 users who have in-house automation systems, such as Traffic and Billing and Agency Stewardship programs which lie at the heart of customers' businesses and are perfectly capable of producing many thousands of paper documents, is to send and receive such documents electronically, transformed before or after forwarding for storage and/or processing on the servers.

15 As stated previously, in a not-so-electronic document, it is core to the very premise of the systems and processes of the present invention to move documents between these automation systems, delivering them in native forms. This will avoid the re-keying time, clerical errors, transport delays, and other issues that challenge customers' current document exchange methods. So wherever possible, documents
20 are sent and received directly to and from to these automation systems without requiring any format or value translations. This begs the question, "How does one move documents from one customer's automation system to another customer's automation system if all we have in between are two client applications that can't do

anything but display documents?" Probably the best way to answer this question is through a real world example.

Harry, a billing operator at a local cable provider, has just finished his reconciliation processes at the end of the broadcast month. Before he does anything else, he wishes to get the national spot bills out to his rep firm. Harry is about to enter that mysterious realm of electronic commerce:

- To begin with, Harry generates electronic affidavits for his national spot trade with his T&B system. It puts all of them for the current period into one big file called an affidavit batch.
- 10 □ Through his client software, Harry logs into the operations center servers.
- He chooses the option to upload electronic affidavits.
- Using for instance the Windows "File Open" dialog, he points to the file which contains the entire batch of electronic affidavits. He presses the "Upload" button. The batch is sent via a file transfer operation directly to the server.
- 15 □ Once the batch is received intact at VNI, a transaction is initiated on the server that parses the batch of affidavits, breaking it down into individual affidavit documents.
- Each affidavit is translated into the common document model core affidavit format and assigned a unique document ID, which will identify it, throughout its life, both to Harry and to the system.
- 20 □ The affidavits are moved into Harry's outbound mailbox, where they are held until he takes further action.

- Based on the data that the server has stored about Harry's electronic trading partners, it is able to assign destinations to all of the affidavits from the batch except one.
- Harry can now view any or all of the documents he just transmitted with his client software. Although it doesn't matter so much to Harry where the documents physically reside, whenever he requests to view a specific affidavit, by selecting it from an on-screen list, it is transmitted to the client application, which then displays it for him. (Because edits are turned off for affidavit documents, he is prevented from making any changes them.)
- Note that the Harry is viewing the affidavits while they reside on the server in core affidavit form, as opposed to a form that is unique to him or any other party. This allows use of a single viewing and editing program for everyone wishing to view affidavit documents.
- In reviewing the documents, Harry realizes that the affidavit for Charlie's Shoes has an error. So he goes back to his T&B system and re-creates an the affidavit for Charlie's Shoes and once again uploads an affidavit batch, except this time the batch contains but a single affidavit for Charlie's Shoes.
- The server is smart enough to know that it can't have duplicate invoices from a single provider so it automatically replaces the erroneous invoice with they newly corrected one.
- Harry is now satisfied with all of the affidavits except one that's missing a target address. Harry realizes it's a new client that the system has never seen before so he selects the target himself from a list of candidate targets.

- With all of them appropriately addressed, Harry highlights all of the outbound affidavits in his viewer and issues a single "Transmit" command for the entire batch.
- The server generates a new affidavit, with a unique document ID, for each affidavit sent by Harry to each target.
- The new affidavits, which are copies of the ones sent by Harry, are moved into the target rep firm's inbound directory, but remain in the core format.
- The server performs a value mapping operation on these new affidavits, which replaces Harry's unique values with the rep firm's unique values. In one case, Harry calls Headline News Network "HDLN" and the rep firm calls it "HLN". The server performs this mapping so that the affidavits look natural when viewed by someone from the rep firm.
- The server moves the original affidavits to Harry's "sent" mailbox and changes their state to "Sent".
- With the sending process completed, Harry is finished, for the time being.

When the rep receives the batch:

- It just so happens that, Susan, the billing manager at the rep firm decides to check their in-box. A large number of affidavits have arrived from Harry.
- After a quick review, Susan decides they look good, as far as she can tell!
- So she highlights all of the affidavits in her in-box and issues the "Download" command.

- The server dutifully copies and translates each affidavit into the format expected by the rep firm's automation system and packs them into a single file (one big batch), and downloads them immediately.
- The affidavit documents are moved on the server from the rep firm's "In" mailbox to their "Received" mailbox.
- Meanwhile, the rep firm's automation system sees all of these new affidavits, in its favorite flavor, and happily consumes them until it sees Charlie's Shoes, where it notes a problem.
- Susan's automations system tells her it likes all of the affidavits excepting one. She returns to her client and acknowledges all but the single affidavit for Charlie's Shoes..
- The server receives the acknowledge command for these affidavits and goes back and mark's each of their originating affidavits from the sender (Harry), as "Acknowledged" and forwards an acknowledgment to him.
- In the meantime, the billing manager issues the "Reject" command against the affidavit from Charlie's Shoes.
- The server promptly responds by creating a "Rejection" document, in the core format, assigns it a unique ID and places it in Harry's in-box.
- When Harry opens his in-box the next time, he sees that his whole affidavit batch was acknowledged except for Charlie's Shoes, there in place of an acceptance, he sees the bold letters of REJECTION.
- Not to worry! He quickly corrects yet another billing error in his T&B system and regenerates the affidavit.

- In his client he chooses to the "Upload" command, pointing out exactly which document he's replacing.
- The system as before uploads it, translates it to core format, and stores it with the same document ID as before. Since it was already targeted to his rep-firm, he rapid fires by once again issuing the "Send" command.
- And the process repeats as before, except this time Susan only uploads a lone affidavit for Charlie's Shoes. Once her automation system has digested the Charlie's Shoes' affidavit without heartburn, she issues the "Acknowledge" command.
- Harry sees that the Charlie's Shoes' affidavit has finally been acknowledged. He takes a deep breath and slowly releases it. He knows that tonight he'll get his first real sleep since the start of billing week.

Prominent Features of Systems and Processes According to the Present Invention

1. Document Centric

One of the underlying themes of systems and processes according to the present invention is the fact that they center on documents, as opposed to X12 transmission batches or application file batches. Users deal better with documents than they do with batches. Group and Interchange IDs mean very little to someone who is trying send affidavits to their rep firm. They just want them to arrive.

When there's a question, as to the payment status of a particular invoice, noone really knows what interchange, group, batch or any other ID it was assigned (other than perhaps the invoice number). However, they do know that it was the

January affidavit for Charlie's Shoes. The approach in giving our customers document management tools, in addition to a communications infrastructure, is unique (and not just in the media world). It is a lot like the overnight delivery services that can track your package's delivery. But the system allows the users to check on it themselves immediately, without having to know any codes.

2. "E-Mail Like" User Interface, but more secure

The client software, in many ways, looks like an e-mail system. It has "In" and "Out" boxes just like an ordinary e-mail system. Users can review any documents they receive on-screen before and after sending or receiving them. They can edit some pieces of information within certain documents when they're in a non-sensitive state. In order to protect the audit trails of interactive information systems, users are not prevented from making potentially compromising edits to certain documents.

3. Group Operations

To make things easier for users, any operations that can be performed on a single document may be performed on groups of documents. So if Harry selects 27 affidavits from his "Out" box, he can press the "Send" button once and all 27 affidavits will be agency (or rep) bound.

4. Custom Document Views

Outside of the normal "In", "Out", "Sent" and other state-based boxes, that are integral to each mailbox, each user can define their own document views. As an example, a billing manager at a rep firm may wish to create a custom document view that shows affidavits from a specific trading partner. A system administrator for an

agency may wish to view all of the documents received from a certain trading partner in the last quarter. Once a user defines a document view, it remains in the system as long as they wish.

5. *Account Management and Administration*

Another unique feature of the present invention's approach is that a trading partner may have many assigned accounts. They may assign each of their users to a different account or have dedicated accounts to a specific purpose, such as receiving affidavits. In this case multiple authorized users might have access to that affidavit account. This allows delivery of orders to a specific buyer within an agency and to still retain the concept that the trading partner is the agency itself, and not a single buyer within the agency.

Each trading partner has at least one user that is assigned as their administrator, who can be authorized to perform administration tasks:

- ☐ Assigning which document types may be sent and received to and from each account.
- ☐ Determine which users have access to which accounts.
- ☐ Assign which document types a user may access within an account.
- ☐ Maintain trading partner lists and cross-references.
- ☐ Update value map cross-references.

In addition, an administrator may create document views that can include documents from any and all accounts, while a normal user will only be able to create views on documents from the mailboxes to which they are assigned.

6. *Comments*

An authorized user may attach any number of public or private comment lines to any document. Public comments are transmitted with the document for viewing by the target users, while private comments are not.

7. *Document History*

All documents can remain on file, although they won't be in the active "Inbound" or "Outbound" state directories once they've been transmitted. They may remain on the server for up to 90 days after they have been completely resolved. Some documents, such as affidavits, may remain on file pending remittance advice indefinitely. The 90-day clock will only begin ticking once remittance advice has been received.

8. *Customer Oriented Translations*

In a typical EDI implementation, each consumer of the electronic data is responsible for translating from the format in which it is received to a format they're able use. On the outbound side (when they're sending documents) they're responsible for reformatting the documents into a standard format that nobody's information system really understands, but is an accepted transmission standard that all of the translation systems can speak (with a lot of help). All of these translations can be costly in terms of manpower, training translation systems, and other areas.

Systems and processes according to the present invention automatically reformat documents when they're downloaded, to a format native to the target's information system. When a user of the system sends documents, they upload them to the servers, in their application's native format, and the servers automatically

translate from that. This way customers do not have to be involved in any translation or deal with its complexities. Customers will receive documents and import them directly into their information systems and export documents directly from their information systems.

5 **9. *Intelligent Value Mapping***

10 In moving documents between trading partners' applications, there is much translation that has to occur, not just in the format of documents, but in their content as well. In normal EDI implementations management these content translations, or Value Mappings, are automatically and intelligently performed by the system. As an example, Harry calls The Cartoon Network, "TOON". Susan calls it "TCN". If Harry and Susan were exchanging electronic documents, using normal EDI methodologies, each would have to know what the other calls it and each would have to perform a value mapping on incoming documents. Systems according to the present invention, which use the common document model, know what each calls
15 The Cartoon Network and can automatically translate between them. So even though Harry sends an affidavit that says "TOON", when Susan receives it, it will say "TCN".

20 **10. *Data Synthesis, Filling in the blanks***

 Few software application systems carry all of the key data elements that may be required for another trading partners' systems. Therefore, in traditional EDI, the people who are creating the translation maps have to fill in a number of holes, and in some cases they have to fill in the holes manually. There have been instances where data required by agency systems doesn't exist in the station's application

systems. This data must be either be captured or synthesized before sending it on to the target.

Sometimes these missing data elements have been supplied somewhere in the document exchange cycle. However the data elements aren't always captured by the trading partner's system because there was either no convenient place to put it, or somebody forgot to key it and their system doesn't tell them they forgot.

For example, suppose Harry, a trusty operator at a station, receives a contract, from a rep firm, for a national advertiser. The rep firm has included the agency's estimate number on their printed contract. When entering the contract into their T&B system, Harry has no place to input the agency estimate number, so he leaves it off. Three months later, when Harry sends a paper affidavit (through the snail mail system) back to the rep firm, it's missing the estimate number.

Susan is concerned, being the billing manager at the rep firm, because she has to lookup the agency's estimate number, from the contract that was sent three months ago, just so she can send a paper invoice to an agency that includes their estimate number.

The instant systems allow everyone to be happy because when the original contract is forwarded to Harry, the estimate number is captured and saved. When Harry confirms, to the rep firm, that he's accepted the contract, his contract number is captured and tied to the agency's estimate number. Three months later, when Harry sends an affidavit, the system will automatically place the estimate number on Harry's affidavit.

11. *Intelligent Addressing*

Just like an e-mail system, some documents need to go to more than one place. Systems according to the present invention can support as many as 256 targets or more for a single document. When the user initiates the "Send" operation
5 it sends a copy specifically suited and value mapped for each target.

Even more impressive is the ability to know the targets of each document when they're uploaded. When a user uploads a batch of documents from their information system to the master platform servers, in much the same way that it performs value mappings and data synthesis, the servers address the documents
10 appropriately. They store tables of each sender's trading partners along with cross references to their keys for those trading partners, and automatically assign targets to each the documents. The user always has the option of changing or removing a target, and when they do so, it updates the server's cross references.

For example, Susan uploads a batch of contracts destined to various stations
15 around the country. The system knows that Susan's information system calls Joe's Cable in Peanut, Georgia - station 17546. So when the system sees a contract from Susan destined to Station 17546, it immediately assigns Joe's Cable as a target. Susan can still add another target, or direct it to a different one altogether, before she sends it.

20 12. *Automatic Document Validation*

Usually, in EDI, when a document's target is picky about the way they receive their data, the sender has to be very cautious about the way that they format and translate their documents before they send them. Thus many EDI shops have

developed complex procedures for sending documents to select trading partners. When this is the case, the instant systems can automate document validation that will notify the sender when they attempt to send documents that violate the target's criteria. The sender can perform a preliminary validation on their documents prior to sending them. In this case the system immediately notifies them whether they meet their targets' qualifications.

13. *Transmission Acknowledgment*

In normal EDI trade, parties must exchange Functional Acknowledgment documents to confirm that the other party received it. Each receiving party is responsible to notify the sender that they indeed received it. Since systems according to the present invention manage the whole exchange, they automatically flag the sender's document when the receiver acknowledges it. The receiver acknowledges or rejects a document with a simple mouse click or keystroke. In this case the sender gets an almost tactile feedback, they don't have to sit and wonder if and when they'll receive an acknowledgment.

The states of their documents are immediately updated as soon as the receiver presses the "Acknowledge" button. By visually scanning the contents of their "Sent" box (state directory) they know which documents have and have not been acknowledged.

14. *Automatic Document Generation*

In a normal exchange of documents between media trading partners, most people have to work through their primary information system, which create

documents that can be exchanged. In some cases this is time consuming and inconvenient.

Let's use the example of Harry receiving a contract from a rep firm. He must first key the contract into his T&B system, and then print a paper confirmation, and then mail it back to the rep firm. On some EDI systems, he no longer has to manually key his contracts from the rep firm. But his information system isn't capable of producing electronic confirmation documents. So, in this case, he's still stuck with his printing and mailing routine.

Systems according to the present invention can auto-generate a confirmation for him from the contract he received. So instead of printing and mailing, he brings up his "Received" box, on the EC@VNI Client, and highlights the contract from the rep firm. After pressing the "En Confirmation" button, he is prompted for his contract and advertiser numbers. As soon as he enters them and presses "Send", an electronic confirmation is on its way back to the rep firm.

In this example, a confirmation is sent in response to a contract without touching an in-house information system directly.

15. Transaction Integrity

All data processing systems experience some amount of down time. In a robust data processing system, downtime is tolerated through redundant fail-safe systems. Such is the case with systems of the present invention. In addition to hardware based redundant systems, the software is built with the concept of a transaction rollback. Should the system fail mid-stream in carrying out an operation on a document, on re-initialization it returns all documents, and their attendant data,

back to their state prior to system failure. Operations can then be re-invoked without any residual effects.

16. *Security*

All documents moving to and from systems according to the present invention are protected through the latest in secure systems technology. All data is encrypted as it is sent and received.

17. *In summary, a way better than EDI*

It is clear from this example that the present invention provide robust business communications and document management systems that literally will extend the capabilities of customers. The days of printing, stuffing envelopes, postage meters, snail mail, and redundant data entry are nearing a close in the media world.

Platforms

1. *Clients and Applets*

The client software according to a preferred embodiment of the present invention is implemented using applets or an otherwise thin architecture, such as Java applets or successors. The driving forces behind this decision are:

- Instantaneous compatibility to virtually every client operating system is one of Java's more attractive features. Eliminates need to force customers to change computing platforms.
- Java's ability to run with applets uploaded to a web browser eliminating the need to distribute the client application using media such as diskettes, CD-ROMs, etc.

- A rich graphical interface support allows creation of applets that run from a web browser but aren't limited to the scant functionality of an HTML type language. They can literally look, feel, and act like typical Windows or Mac applications, with windows overlaying windows rendering more 3D, rather than 2D, performance.
- 5 □ Java readily and simply supports data communication over networks.
- Being an object-oriented language, Java permits ready creation of reusable classes for other applications.
- It is a widely supported, rather than a proprietary language dependent on single vendor. There are also numerous Java programmers to be found with those
- 10 ranks growing daily.

2. *Servers and UNIX*

The server operating system of choice is preferably implemented in UNIX due to the following:

- Existing electronic commerce applications run on the UNIX operating system.
- 15 □ UNIX is supported on some of the more powerful computing platforms in the world. It offers scalability, fault tolerance, and throughput that will be required of this industrial-strength application.
- Programmer ubiquity.
- UNIX has built-in and robust support for all of the TCP/IP based communications.
- 20 □ Multi-tasking is something UNIX was designed to do from its inception.
- Out of the box UNIX sports massive amounts of text processing utilities and scripting languages that prove very useful in electronic commerce applications.

3. *Applications and Object Oriented Languages Such as C++*

The core modules in the master platform server system, such as the Document Manager and Value Mapper objects, are preferably rendered using C++:

- Given the potential to be connected to hundreds, if not thousands, of clients, the core objects in the server application require as much throughput as possible. C++, when skillfully written compiles to very fast native executing code.
- C++ is very well supported on all UNIX platforms.
- There is an abundance of C++ programmers.
- C++ connects very well using the UNIX communications facilities.
- It is possible to readily hook up to SQL databases using very standard ODBC drivers or even embedded SQL. It's probably better to use ODBC but in cases where speed counts it is possible to fall back to embedded SQL should the situation warrant it.
- C++ has excellent support for reading and writing with streams on local devices.
- C++ is a very object oriented language, which allows creation of readily reusable classes.

4. *Databases and SQL*

The database tables for the server application preferably reside on a Sybase SQL system because:

- Sybase is supported by powerful UNIX systems giving much-needed performance.
- It readily supports ODBC and drivers for it are widely available.

- Sybase has an excellent reputation for support and have remained a front-runner in the highly competitive SQL database market for many years.

5. *Client/Server Communications*

For the foreseeable future, communications between the client and the master platform servers will take place over TCP socket connections. Should it prove necessary in the future to implement load balancing and other advanced features, it may be necessary to use something that layers over TCP Sockets, such as CORBA.

Server Issues

Although from the foregoing one could easily gain the impression that the master platform servers are a substantial application, the core of the system is nevertheless not providing all of the functionality heretofore expressed.

To illustrate, the list of tasks necessary to move a document between trading partners can be short or long, depending upon the requirements specific to each information system (or lack thereof). In some cases almost nothing has to be done, excepting moving the document from the sender to the target(s), while in others, it may be necessary to perform extensive validation, complex translations, large batch collations, etc. It is therefore difficult to create a far-reaching and uniform structure that will efficiently handle all of these different document management and exchange scenarios. So to jump to the heart of the matter, systems and processes of the present invention don't automatically pursue that option.

Specifically, the server system is preferably a skeleton application that provides a number of core services, like communications and document tracking, but

the validations, translations, collations, and even document routing come from smaller individual applications that, in essence, put the meat on the skeleton. The server provides the glue to tie these sundry applications together into a cohesive framework.

5

Storage Strategies

All documents are preferably stored persistently on the server. No documents need reside at the client except temporarily for editing and viewing purposes. The server preferably performs all operations. Even when a client is editing a document, it need only be saved at the server.

One of the overwhelming concerns in building the server system is throughput. Given the potential of many hundreds of users and perhaps hundreds of thousands of documents, the server must perform all operations as efficiently as possible. On the other hand, database managers (DBMS) are wonderful tools that can allow ad-hoc queries against all kinds of data, but in utilizing them a high price is paid in performance, setup, maintenance, and accessibility.

Where data needs to be stored persistently, the present invention prefers generally not to store and organize it in the DBMS unless it meets one following criteria:

- The primary data object is of a global nature, in that it can't be organized systemically within a single trading partner, mailbox, and/or user. An example is a document, a document ID needs to be assigned to each document globally so that one can refer uniquely to each. However, the entire contents of a document do not really need to reside in a database. Another example, a user needs to be

defined globally because they will each have their own login ID, password, etc.

Both of these data objects share a global context where as a document view, is

only used within the context of a single user. One can therefore organize

document views under the context of a single user and have persistent file-based

storage designated exclusively per user.

- The data will likely be needed for ad-hoc queries where the queries have the potential of global context.
- A relationship to other data objects needs to be enforced that can not be easily enforced without using an onerous and proprietary scheme.

Form Documents – Common Document Model

All documents, excepting transmission batches, are preferably stored on the server in a standard format, according to a common document model. Each

document type that moves through server has its own standard format. When an incoming transmission batch is received at the server, it will immediately broken

down into individual documents and stored in their format. Preferably, each form document will occupy a single file, each file will hold but a single document. Only document transmission batches, that have been uploaded to the server or are to be downloaded from the server, may contain more than a single document per file.

Whenever a client requests a document, it is forwarded in its standard form.

Standard form documents are always translated to and from.

The form of any document type contains all of the data elements required by all trading partners for the same type. So the form of a document type contains a superset of the data elements required by any single trading partner. The form of an

affidavit, for example, contains all of the data elements required by industry standards and various industry players. Therefore, from a standard form affidavit it is possible to derive an affidavit satisfactory to any of these automation systems.

Form documents are also mappable. This is to say that they are organized into records; identified by the Record ID residing in first data element of each row and delimited by the newline ('\n') character; and data elements (fields), variable length delimited by the pipe symbol '|' within each row (record). So it is possible to refer to any specific record by its ID and any data element by its Record ID and position, i.e.; Record XYZ, Position 6.

Standard form documents preferably contain only ASCII characters, no non-printable characters, no binary data whatsoever. All floating point numbers, integers, BCD, etc. will all be converted to strings.

Several benefits will result from this approach:

- The client only has to contend with a single format for any document type. This cuts way back on the necessary development for the client software.
- It reduces the number of translation engines that one has to create and maintain.

To illustrate, suppose there are affidavits coming from 3 different automation systems, A, B, and C. There are 3 different target automation systems for outgoing affidavits, X, Y and Z. If there is no core document format, one would have to maintain translations for A to X, A to Y, A to Z, B to X, B to Y, etc. or 9 translations. Then add another affidavit source and destination for a total of 4 in each direction. There are now have 16 separate translations to maintain. If one trading partner changed their format slightly, one would have to update now 4

translators for one minor change! Using a core format with 4 senders and 4 targets, one has only 4 incoming translations and 4 outgoing translations for a total of 8. If one target changes their format slightly, only a single translator need be updated.

- 5 □ Given that the standard form is mappable, one can create generic document utilities that function in a table driven fashion for any trading partner. For example, one can build a value mapper that will replace the sender's values with the target's based on simple coordinates. So to change the network names for a document, one could tell a program that is has to swap networks values around,
10 it could know that for affidavits, the network values are found in record 51, position 10. It can then in a network cross-reference table see which values each partner uses and quickly swap them. If one stored affidavits in 4 distinct formats, one would have to have 4 separate programs just to swap network values.
- 15 □ Documents will be editable with any text editor or and even viewable with a simple screen dump.
- 20 □ It is possible to refer to and programmatically operate any specific data element within a document without incurring the overhead of storing them in a database. They can still be retrieved quickly from the disk using lightweight streaming mechanisms. One may still store select key elements in a database to have the benefit of executing queries against the key elements.
- It is possible to manipulate the documents with simple shell script programs.
- Documents in this form will translate more readily to X12, EDIFAQ, and other standardized EDI formats using off-the-shelf translation software.

Standard Document Record Types

One or more of the following record types may appear in a standard form document according to a preferred embodiment of the present invention. These record types are consistent across all types of documents. Each provides
5 information that may be used system wide.

- **Document Header (00)** – Each form document stored on the system will begin with a document header record that contains the following elements:
 - **Document ID** – The unique ID that identifies each individual document.
 - **Document Type** – States the type of document such as an invoice, order, confirmation, etc.
10
 - **Version** – The version number of the document type.
 - **Trading Partner ID** – Identifies the trading partner owner of the document.
 - **Mailbox** – Contains the owner mailbox name.
 - **User** – If a user that owns a document, this will contain their User ID.
 - **Origination Date/Time** – Indicates what date and time the document was
15 received at or created by the server.
- **Target (02)** – A document may contain from zero to any number of Target records. These identify to whom a document will be sent. Each target will identify a single document destination.
- **Source** – Indicates the source of the target. May contain the word “Auto”
20 indicating that the target is a result of the Auto-Target process. Can also contain a User ID indicating who added the target.

- **Trading Partner ID** – Identifies a destination trading partner.
- **Mailbox** – Contains the ID of a destination mailbox.
- **Trading Partner Name** – Contains the name of the targeted trading partner.
- **Send Date/Time** – Indicates the date/time when a document is actually
5 forwarded to the target. This element remains blank until the document is
sent.
- **Comment (03)** – A comment record is always created directly by a user. These
are for viewing purposes only and don't constitute usable data.
 - **Send** – Indicates whether the comments should be included when they are
10 forwarded to their target destinations.
 - **Text** – Contains the text of the user comment.
- **Message (04)** – Messages are placed in a document by the system when some
event occurs where a user needs notification of some event.
 - **State** – Indicates whether or not a message is critical. (Documents may not
15 be sent so long as there are critical messages.)
 - **ID** – May contain a message ID for standard messages.
 - **Text** – Contains the text of the message.

Document Definition Files

The structure of each form document according to a preferred embodiment of
20 the present invention may be defined in a tabular form and stored in Document
Definition Files (DDF). Each will dictate the structure of a single version of a form

document. This structure allows document structures to evolve and still provide usability of older versions of documents. Each time the form of a document is altered, a new DDF must be produced so that the system utilities can decipher its contents.

5 A case in point, a trading partner may have many order documents archived at any given instant. Older order documents may be of a different vintage than newer ones. Because system document processing utilities, such as translation programs and screen forms, interact with documents indirectly through a mapping class that decipher the documents per the instructions of the DDF, the user may be
10 completely ambivalent about the operations they may perform against which version of document. Even the document utilities are blind to minor changes in document structure since they don't directly interact with the document files themselves, rather with objects that provide a level of indirection between the physical document files and themselves.

15 Each DDF preferably contains the following information regarding a specific version of a document type:

- The types of records contained in a document.
- The ID of each record type.
- The nature of each record type (Optional, Required)
- 20 □ The relative hierarchy (looping structure) of the record types.
- The name of each data element in a record.
- The position of the data elements within the record.

- The type of each data element (Numeric, Alpha, Alphanumeric, Date, Money, etc.).
- The maximum and minimum size of each data element.
- The nature of each data element (Mandatory, Optional, Conditional)

5 By convention, DDFs reside in a special directory and will be named according to the document type and versions they represent. Their filenames can be structured as followed "TTTTTTTT.VVV.def" where 'T' represents the characters of the document type and 'V' the digits in the version of the document type. Thus a typical document definition filename might be "order.001.ddf".

10 ***Document Key Files***

As stated in previous paragraphs, it is key to throughput and accessibility strategies to keep our documents in ASCII flat files. However, this poses the problem as to how to support the need for ad hoc queries against documents by clients and the easy collection of their key values for client presentation.

15 On any given document type, there will not be very many data elements which would be considered as a distinguishing characteristics, or "keys", and be used conditionally within an ad hoc query. Take an affidavit for example; some distinguishing data (key) elements might be the invoice number, advertiser name, invoice date, agency name, station, etc. But almost noone would be interested in
20 conditionally querying against or viewing in summary the 1st agency address line, 2nd address line, agency-commission, network, etc.

Systems according to the present invention thus preferably support both lightweight document storage and heavyweight ad hoc queries by identifying the key

data elements for each document type and storing instances of those key elements into a database table.* Taking this approach, one does not have to create a special table for each document type in the database. Instead, one can store the key element instances with a purely table driven strategy.

5 The strategy involves defining which data elements are "Key" to each document type in a Document Keys File (DKF). The key data elements are listed by name, one per line, in the DKF. The names must correspond to the data element names as defined by the DDF. Since the data elements considered "Key" for a document type will transcend versions of document types; and since database
10 storage of key elements has moderate implications, the list of key elements for each document type will be stored in separate tables (files) from the document definitions (DDFs).

 With these instances of key elements defined in a DKF and the actual values stored in a database table, it is very simple operation to perform ad hoc queries
15 against them. It also avoids weighing down the system with complex, large, cumbersome, and curiously intertwined database tables that, aside from providing document storage, represent complex data relationships embodied within each document. Thus, when a request to build a customer view of orders for "Johnny's Agency" is forwarded by a client, the server will be able to quickly respond by
20 querying the DBMS for key element instances stored in a simple lightweight table. When a client requests a summarized view of the documents contained in a folder, the server can easily determine which are the key data elements for the document type and extract only those from the documents and return them to the client. When

more support is needed for a new document type, it becomes a simple matter of creating a DDF and a DKF for the new document type. This is a simpler process than generating entity relationships and designing and normalizing database tables just to store documents and to identify a few key elements.

5 Only data elements in records of the 1st order (one instance per document) may be defined as keys. Data elements resident in records where there may be multiple instances of that record type in a document, and therefore multiple instances of a key within a document, can not constitute a distinguishing characteristic of a document. For example, a 'Network' data element in an affidavit resides in the
10 'Schedule Line' record. There can be many instances of 'Schedule Line' records in an affidavit. Therefore displaying the 'Network' of a document is not only ambiguous (because there can be many of them) but also not a distinguishing feature of any affidavit.

Standard Mailboxes

15 Each trading partner on systems according to a preferred embodiment of the present invention can be configured with several standard mailboxes. Each has a specific purpose and is not optional, configurable, or capable of being renamed, as far as the client is concerned (at least initially). Documents may be moved between mailboxes as various operations are performed against them.

Concurrent Document Access

20 Given the simultaneous multi-user nature of systems according to the present invention, it is imperative to allow simultaneous access to documents by multiple

users. But there must also be protection of the integrity of the documents themselves by disallowing potentially conflicting operations to be taken against them. For example, two users may nearly simultaneously retrieve the same document. Let's say the 1st user makes some changes to the document and then stores back to disk at the server. The 2nd user may wish to send the document after viewing but some changes have taken place since he reviewed the document. This has the potential to perform an unwanted operation.

The issue can be resolved through the use of time stamps. When any operation is invoked against a document by the server, whether it affects the document store or not, it touches the document file so that it is time stamped. Whenever a client retrieves a document, the server attaches an ID record at the start of each document that contains its ID, its location, and the time stamp on the document file when it was read. If any operation against the document is subsequently invoked, the client passes back the ID record, which contains the time stamp of its latest read. The server will then compare the time stamp against the document file's current time stamp. If it has changed, the server rejects the operation back to the client informing them that changes have taken place since their last read.

Additionally, whenever an operation is in-progress on a document, the server puts an exclusive lock on the document file so that it can't have any other operations invoked against it. As soon as the operation finishes, the server releases the lock. This approach does not require locking any files for any extended periods of time (only while a transaction is in-progress). It also permits concurrent access by an

unlimited number of users. No database access is required whatsoever, and touching a file is very cheap in terms of system resources. Users can retrieve a document for editing purposes and leave it open on their workstation for hours without hurting the system whatsoever. No complex scheme is required either to notify the users of updates. The users aren't inconvenienced, except in very rare circumstances when a document is changed underneath them. In the very worst case they may have to re-read the document and re-invoke their operation.

View Files

Whenever a user defines a document view, a "<view name>.view" file is created that stores the crucial elements of the query, that in essence, define the view. A second file is created, whenever the query is executed, name "<view>.view" which contains the document ID's of each document matching the query criterion. So if, for example, a user defined a view named "affidavit", the document manager would create a file named "affidavit.query". When the user populated the view, a second file would be created named "affidavit.docs". Both of these files would reside in the user's home directory (EC_ROOT/<tp_code>/<user name>).

Document Operations Manager

One form or major part of the master platform server, can be the Document Operations Manager (DOM). It lies at the heart of the server's functionality by managing all of the communications and connections to client instances and by managing and invoking operations for they request. It also centralizes access to the database.

Transactional Integrity

It is necessary for DOM to insure the integrity of each transaction executed. Since all transactions are script driven and many, if not most, occur mostly outside the context of a DBMS, DOM must insure the relational integrity of the system in the event of a mid-execution failure. Given that each step of each transaction is logged in a transaction log file, DOM can tell which transactions were not properly terminated.

Upon startup, DOM checks for any transaction log files. If any are found, it calls up the appropriate operation from the Operations table, and invokes the specified Rollback Script, if there is one. These Rollback Scripts are responsible for cleanly backing out transactions. When the script successfully returns, DOM will delete the transaction log file itself.

DOM will not respond to login requests, or any other messages, until it has attempted to reverse all partial transactions.

Client Requests Overview

A DOM will receive and operated on several standard requests from a client. DOM will always return the value SUCCESS (and potentially a result set) when it succeeds. It will return FAILURE and a text error message when it fails.

All of the operations are executed natively by DOM (or one of its threads) except the ExecOperation request. These are executed through shell scripts that, by convention, return success or failure values to DOM.

- **Login** – Establishes a connection to DOM.
- **Logout** – Destroys a connection to DOM.

- **GetDoc** – Tells DOM to retrieve the specified document(s) by its ID.
- **GetShortDoc** –Asks DOM to retrieve an abbreviate version (key data elements only) of the specified document(s).
- **StoreDoc** –Requests that DOM re-write the passed document.
- 5 □ **AddDoc** – Asks DOM store a new document.
- **DeleteDoc** – Tells DOM to delete the specified document.
- **Move Doc** – A request for DOM to move a document to another mailbox or state directory.
- **GetKeyNames** – Requests that DOM return the key element names and data
10 types associated with the passed document type. This information will serve as an aid for helping users to create custom document views.
- **BuildView** –Asks DOM to build a document view, under the specified name, and return the key elements of the resultant documents.
- **GetView** – Instructs DOM to return the key elements of the documents
15 associated with a particular view.
- **GetQuery** – Allows the client to retrieve the contents of a query that formulate a document view.
- **StoreQuery** – Asks the server to store the contents of a document view query under the specified name.
- 20 □ **RegisterDoc** – A request for DOM to register a Document (assign it a Document ID) that is stored under the specified name in the user's upload directory. This is normally invoked on transmission batches that were previously transferred via FTP to the upload directory.

- **GetTargets** – Asks DOM to return a list of authorized targets for the specified document type, so the client can have the user select them from a list.
- **AddTarget** – Makes DOM add a target to a document.
- **DeleteTarget** – Tells DOM to delete a specified target from a document.
- 5 □ **GetOpsList** – Requests DOM to return a list of the operations supported for a specific document type.
- **GetFolders** – Returns a list of folders available to the specified user.
- **ExecOperation** – Instructs DOM to execute a stored operation on the document(s) specified by the passed Document ID(s). Examples of stored
10 operations would be commands like “Send”, “Validate”, etc. DOM depends on scripts, registered as valid operations in the Operations table in order to carry out the requested operation.

Message Format

Request and return messages use the same format and are variable in length,
15 with 16 bytes required, 12 in the header and 4 in the tail. All messages will use the following format:

- **Client ID** – 4 bytes, offsets 0 - 3
- **Request or Return Value** – 4 bytes, offsets 4 - 7
- **Bytes in Buffer** – 4 bytes, offsets 8 - 11
- 20 □ **Buffer** – Length specified by Bytes In Buffer (offsets 12 – n)
- **End of Transmission (EOT)** – 4 Bytes (immediately following buffer)

Both client and server retrieve packets in two steps. In the first step, they retrieve the first 12 bytes of the message, determining if the request or return value is valid and getting the length of the buffer. Once they know it's a valid message, they will go ahead and retrieve the buffer and EOT. They store the buffer contents in
5 allocated memory, parse it into its components, and process it.

In request messages (server bound) the 2nd element holds the request value, while in result messages (client bound) it contains a request return value. The return value can be zero, or SUCCESS, a positive integer indicating a critical error, or a negative integer indicating a non-critical error. On any error return, critical or not, the
10 buffer always contains a text error message describing the condition.

In any foreseen cases, the Buffer would always contain ASCII data exclusively. When numerous data elements are included as part of a request packet (server bound), they will be pipe ('|') delimited. In result packets (client bound) data elements are also pipe delimited. But often result packets will store result sets in
15 buffers that will include not only data elements, but whole records and documents as well. In these cases, the buffer will have newline ('\n') delimited records, and form feed ('\f') delimited documents. EOT will always signal the end of the buffer, as well as the entire message.

Script-based Operations

20 As alluded to previously, most client requests are executed internally and directly by DOM or one of its threads. This works very well for limited and very concise operations, such as storing and retrieving documents, building document views, etc. But there are so many operations that may need to be performed against

a document that it is difficult to create a special request message for each.

Additionally, there are some operations such as translations at least partially using 3rd party products to perform them. Nor can one anticipate every operation that may have to be performed against each document type for even a single trading partner.

5 When one factors in the number of trading partners to be served, the potential operational diversity is enormous. It is expected there will be significant writing of many custom programs to satisfy various large trading partners' unique requirements. Finally, the system will have a number of different people working to create these custom solutions and it is desired to give them as wide a latitude as possible within the confines of auditability, transactional integrity, and good form.

The best way to satisfy all of these requirements is to utilize the scripting languages native to the UNIX operating system. But these custom scripts must be tied together into a cohesive framework that provides support for the diverse operations. Thus DOM supports a special client request called "ExecOperation".

15 The client tells DOM which script to execute by specifying an operation code. All major operations that translate and route documents will be executed through one of these shell scripts.

The scripts are executed by a DOM service thread, but not natively with compiled C++. It actually invokes an instance of a shell and instructs it to execute the script.

20 The DOM thread will start a transaction in the database and create a log file that will help it to determine whether a transaction, executed via a script, has successfully completed. Each script program has a reversing counterpart (rollback) script that

can undo a transaction that is partially completed by its associate script. No script may be registered as a valid operation without a corresponding rollback script.

A script is registered in the system when it is called out in an operation record (Stored under the Operations table by the DBMS). Operation records are unique to each document type. In other words, one may have a "Send" operation for 20 different document types, but there may be a different script associated with each. There is no limit to the number of operations that can be assigned to a document type. Each operation names a single execution and rollback script. The execution script is called when the client issues an ExecOperation request naming the operation. The rollback script is called exclusively DOM on startup, if there is an incomplete transaction lurking in the system. Each script is responsible for updating the transaction log file so that its rollback counterpart can do its job properly.

The client is responsible for passing the appropriate script arguments to the server so that it can in turn pass them along to the script. There is no way feasible for DOM to be aware of the number and types of arguments required by a script, so it will be the client's responsibility to supply them.

Request Messages

Request messages are sent by client instances to the Document Operations Manager requesting some action be taken. The first message sent by a client must be a login request that establishes a connecting socket through which the client and server (DOM) will subsequently communicate. The client is responsible for

terminating the connection when has completed operations using the Logout request.

On all requests except the Login and Logout requests, a DOM thread that is manning the client connection, will always check to see if the current user is authorized to perform the requested task. It will reject requests a user is not authorized to perform.

In all cases following a request, DOM will respond with a result message indicating success or failure. Failure comes in two flavors, critical and non-critical. When a non-critical error is returned, the client should check the returned result set.

Login Request

Before a client instance can perform any operation or view any document, it must first connect to DOM. It does so through a TCP socket at DOM's assigned primary port following these steps:

- DOM receives a login request from a client that is attempting to connect passes its client ID, user name, and password.
- DOM assigns an available port to the specified client ID and registers them both in the port registry.
- DOM spins a new thread assigned to a secondary port passing it the client ID, user name, and password.
- The new thread opens the secondary port and waits on client requests.
- After spinning the thread, DOM returns to monitoring the primary thread for login requests.

- The thread verifies the user name and password against entries in the "User" table.
- If the user name and password are verified, it will confirm a connection to the client by returning the SUCCESS and newly assigned port number, otherwise it will return FAILURE and the thread will follow the Logout procedure (See Logout Request).
- It stores the user name and password to use them for subsequent security validation.
- The connection thread will stay active monitoring the port for incoming requests until:
 - The user logs off
 - The connection has been inactive for more than the time-out period
 - DOM terminates the connection because the same client ID requests a new login.

Logout Request

As soon as this request is received by a DOM thread:

- The thread returns a SUCCESS result to the client with no accompanying data.
- It blocks on exclusive access to the port registry.
- It de-registers itself and the client ID.
- It closes the port.
- Lastly, the thread exits, terminating execution.

GetDoc Request

This request is sent by the client to retrieve selected documents in their integral form. Each document requested is by its document ID that is copied, pipe-delimited, into the buffer portion of the message.

5 Once the message is received and validated and the user is authorized by a DOM thread it:

- Begins a loop operating on each document ID.
- Queries the Document table to determine the storage location for each document.
- 10 □ Reads the document file time stamp
- Builds a header record for it containing the document ID and the time stamp and stores it in the allocated memory buffer.
- Reads document from file and stores properly delimited in the memory buffer.
- Loop ends after the last document ID it received
- 15 □ Builds an appropriate result message with the documents contained in the buffer area
- Sends the result message.

The thread will return SUCCESS value if it succeeds in loading all requested documents. If a single document has been requested and it fails to load the
20 document, it returns a critical error. Finally, if multiple documents are requested and at least one document is successfully loaded, it returns a non-critical error.

GetShortDoc Request

This message is used by the client when it needs to present an abbreviated view of one or more documents and it knows their document ID's. The server will return a result set containing only the key element instances of the specified documents, i.e. invoice numbers, advertiser name, agency name, etc. The client will populate the request buffer with pipe-delimited document ID's for all of the desired documents.

After the server receives and validates the request message and checks the user's authorization, it formulates a SQL Query that includes all key element instances for the desired documents. After invoking the query it retrieves the elements and copies them with their document ID into a memory buffer. In following conventions, it will delimit the key element instances from the document ID's with pipe symbols ('\'), key rows with newlines ('\n'), and documents with form feeds ('\f').

After populating the buffer, it builds a result message and returns it to the calling client. The message will return a SUCCESS value only if it is able to load the keys from all requested documents. It will return a non-critical error on partial loads, and a critical error when it fails to load keys from any of the requested documents.

StoreDoc Request

A client will make this request when it wishes to save one or more preexisting documents after changes have been made. The client will build a request message containing the whole, delimited contents of each document it wishes to store. It must include public and private comment records because any previously existing comment records will be overwritten. Each document must start with a valid ID

record containing the server assigned document ID, file location, and the timestamp of the client's last read. Once it has appropriately built the request message, it will forward it to a server thread.

When the server thread receives and validates the request and checks the user's authorization, it will perform the following on each document:

- ❑ Compares the timestamp the client passed, in the ID record, to the current time stamp on the document file. If it has changed, it returns a critical error message informing the client that the file has changed since their last read.
- ❑ It then attempts to issue an exclusive lock on the document file. If it fails, it returns a critical error to the client because an operation, against the document, is in-progress.
- ❑ An attempt is made to write the file to its appropriate location. If that fails, it returns a critical error.
- ❑ It calls the key extractor that will locate its key elements and update the KeyInstances table in the database.

Once the server successfully completes the previous steps for each document, it builds and sends a return message indicating success.

AddDoc Request

A message a client issues when it wishes to store a single brand new document in the system. The client will prepare by building an ID record containing the target mailbox and state directory. The remaining elements of the ID record should be blank. The server will populate them after it stores the document.

When the DOM thread receives the request, it takes these actions:

- It inserts a new document record in the Documents table. The DBMS will create a unique ID for the document. Failure to insert the document record in the table constitutes a critical error.
- It saves the passed document in a file with the same name as the new ID and in the mailbox/state directory location specified by the client.
- It calls the key extractor to update the KeyInstances table with key elements from the new document.
- It builds a return message that includes the ID record, now populated with the document ID and timestamp of the document file.
- The return message is sent back to the client indicating success (assuming everything went according to plan).

DeleteDoc Request

In this message the client must pass an ID record containing the document ID and the timestamp of when the client last read it.

The DOM thread that operates on the delete request will only do so if the user is authorized to delete documents from the mailbox. Once it receives the request, it attempts to place an exclusive lock on the file. If it fails it returns a critical error indicating that an operation against the document is in-progress.

Once it successfully locks the file, it deactivates its pertinent records in the database. When that has successfully completed, it moves the document file to the deleted directory for the mailbox and returns a message to the client.

GetKeyNames Request

The client uses this message to retrieve the names and data types of the key elements for the specified document type. When the client needs to set up columns in a viewer or when it needs to formulate criterion for a document view, it can use
5 this call to retrieve key column names for a document type.

When a DOM thread sees and authorizes the request, it queries the DocTypeKeys table for the names and data types of the key elements for the selected doc type. It then pipe delimits the elements, stuffs them in the buffer of the return message. Unless DOM is unable to query the database, it will return
10 SUCCESS.

BuildView Request

The client uses this message to build custom document views for a user. The message buffer must contain the document type and the key criterion used to build the view. For example, "AdvertiserID = 10035 AND InvoiceDate >= '10/31/97'". The
15 criterion section of the message buffer must form a valid "WHERE" clause of a SQL expression. The server will complete the other portions of the SQL expression such as the "SELECT" and "FROM" clauses.

After receiving and validating the message, the server thread does the following:

- It builds a SQL statement filling in the portions the client didn't in order to get a
20 result set back from the DBMS that contains all of the document ID's that meet the client's criterion.
- It then issues the SQL statement to the database and retrieves the result set.

- It creates the view files with name "<view name>.query", that will contain the query text, and "<view name>.docs", which will hold the list of document ID's.
- It then walks through the list of document ID's, identified by the SQL query, and copies their key elements, appropriately delimited, into a message buffer.
- 5 □ It then returns the message buffer with all of the records in the client.

The reason that view files are kept is so that a client can later just look at the view of the documents without having to re-query the database. The client can also retrieve the query text and view it should they wish.

GetView Request

10 A client will send this request when either a view has previously been created, in which case it will supply a view name, or it wishes to retrieve key elements for all documents and a given state, in which case it supplies the mailbox and document type. In either case, the server will return a result set that includes key element instances for all of the documents named by a view or state/doc type.

15 After receiving the request, it will determine if it is dealing with a view. If so, it will open the "<view name>.docs" file and retrieve the key element instances for all documents contained therein. If the client has requested for a state/doc type combination, it will look at all of the documents sitting in that location and retrieve the key instances for them.

20 After retrieving the key instance information, it will pack them appropriately delimited into a message buffer returning the result to the client.

GetQuery Request

This message allows the client to retrieve the query text associated with a particular view. The client will put the view name in the buffer portion of the message.

- 5 When the server receives the request, it simply retrieves the contents of the file "<view name>.query" and place it in the buffer portion of a return message.

StoreQuery Request

- 10 Provides the means for a client to update the contents of SQL query related to a particular view. In this case the client places the view name and the entire next of the updated query in the buffer section of a request message.

Upon receiving the request, the server will attempt to gain an exclusive lock on the file "<view name>.query" and rewrite it with the updated contents.

RegisterDoc Request

- 15 Typically a client will upload transmission batches, they wish to process, into the upload directory belonging to their user. DOM will not process a transmission batch until it has been registered in the system as a document. This facility allows the client to register a batch, once it has been uploaded. The client must initialize the buffer portion of the request with an ID record containing the filename, document type, mailbox, and the state directory in which it should be stored.

- 20 After receiving the request, the DOM thread will try to find the file and move it to the appropriate directory. It will then add a row to the Document table, which will create a unique document ID. Then it will build a return message where the buffer

contains the ID record completed with the new document ID. After the client receives the message, it will then be free to invoke qualifying operations against it.

GetTargetList Request

A request a client makes when they are trying to present a user with a list of possible target trading partners for a document type. In order to send this request, the client must pack the buffer section. It must contain the document type being addressed, a qualifying trading partner type, and an indication, as to whether it wants all potential trading partners, or just the ones it has targeted previously.

When the server receives the message, it will either query the TPXRefs or the MBDocTypes table for potential trading partners and mailboxes. It will query the TPXRefs table if the client only wants to look at trading partners it has used in the past. Otherwise it will query the MBDocTypes table to get a list of every potential trading partner and mailbox for the specified doc type. When the results are returned from the database, it packs them in message buffer and sends them back to the client appropriately delimited.

GetTargets Request

A client makes this request when it needs a list of the targets assigned to a specific document. It must pack the document ID into the buffer portion of the request message.

The server will query the DocTargets table and respond with a complete list of all the targets selected for the document including the tp_code and mailbox for each.

AddTarget Request

With this request, the client instructs DOM to add a target to the specified document. The client must pack the document ID, target trading partner, and target mailbox into the buffer portion of the request.

- 5 After receiving the request, DOM will attempt to insert the desired target into the DocTargets table.

DeleteTarget Request

Using this message, a client may request that the server delete the target, specified in the buffer section, by its tp_code and mailbox.

- 10 Once DOM receives and parses the message, it will attempt to delete the selected target from the DocTarget table.

GetOpsList Request

Allows the client to download a list of the script-based operations supported for a particular document type. The client will use this list to know which operations are legal for a specific type.

15

DOM will respond to this request by querying the Operations table and returning all of the valid operations for the document type specified by the client.

GetFolders Request

- 20 Returns a list of the folders (state directories/doc types and user defined views) available to the specified user. This is so that the client can display the folders available to each user.

ExecOperation Request

The client will pack the buffer portion of the message with the document ID's and desired operation to be invoked. The client must also pack any arguments required by the execution script.

5 When the server receives the message, it will unpack the document ID's and invoke the script indicated by the operation code for each.

Server Utilities

10 These objects all perform tasks that can be somewhat standardized across multiple document types and trading partners, but can't be executed consistently in any particular order. Therefore, the individual scripts that call them control their use. There may be many more utilities developed over time that are within the scope of the present invention. Many of them will be rather specialized, where these are all more generic in nature.

15 For the present, all server utilities are preferably implemented as command line executables. Later, a scheme may be devised where they become daemon processes, where they always stay loaded in memory, and control their operation through a socket or another IPC mechanism.

Document Router

20 This object is designed to move, or copy rather, documents from a source mailbox to a target mailbox. It takes arguments for document ID and state.

It utilizes records stored in the DocTargets table to determine which targets a document should be routed. Once it has a list of all the targets, it performs the following steps for each:

- Creates a new copy of the original document, excluding any private comments.
- 5 □ Registers the document by adding a document record to the Documents table. The table will assign a new document ID.
- It saves the contents of the new document under the target trading partner, mailbox, and state (passed as an argument).
- 10 □ It calls the Key Extractor to update the KeyInstances table with key elements from the new document.
- It will then attempt to insert a TPXRef record into the TPXRefs table for the trading partners and mailboxes associated with the current exchange (TPXRef instances are used to help trading partners identify with which other trading partners they actively trade.).

15 ***Translation***

Even though there is no single, generic, translation utility, or any convention for the arguments they will take, then need to be mentioned. Many of the translations will be performed by GenTran Server components, while others will be custom executables or script programs. Some will translate from an application form
20 to VNI form and vice verse. Yet others will move from X12 to VNI and from VNI to X12. They will be invoked via scripts and they will read documents in and push them out the other side in different form. Translation utilities will normally be called by

upload and download scripts which will invoke them while moving documents in and out of the system.

Transmission Batch Parsers

Transmission batch parsers are nothing more than translation utilities that
5 converts multiple application form documents in a single file to single VNI form documents in multiple files.

Value Mapper

The value mapper utility takes arguments for a document, its source trading partner, and target trading partner. Its purpose is to change values from meaningful
10 to the source into values meaningful to the target. It depends on the ValueXRefs table, to tie the values together, and on the DocTypeKeys table, to tell it which data elements need to be mapped.

The ValueXRefs table ties each trading partner's values to VNI values, which represent a central standard. This way requires maintaining fewer value
15 relationships. The ValueXRefs table is updated programmatically by various translators and parsers. It will also be updated manually by our system operators. The DocTypeKeys table identifies the elements of a document type that potentially need value mapped. It can be maintained manually whenever a document type is added or updated within the system.

20 Once invoked, the value mapper will query the DocTypeKeys table and build a list of those elements that need mapped. It will then walk through the list and for each element perform the following steps:

- Retrieve the current element value from the position identified by the DocTypeKeys record.
- It will look up the cross-reference for the element based on the current value.
- With the cross-reference record it knows what VNI calls the element value.
- 5 □ It will then look up, through VNI's value, the target's equivalent value.
- It replaces the current element value with the target's value.

Once it is through processing all of the data elements, it will rewrite the contents of the document.

Auto Addresser

10 This utility is charged with attempting to address, find targets for, a document based on past trades with trading partners. When invoked against a document, it will lookup past trading partners based on data elements that have been identified as trading partner names. It will then look, in the TPXRefs table, for a trading partner that has a matching name. After finding a match for the document, it will add a
15 target for it.

Data Synthesizer

This class of utility is far from generic and will implemented specifically for a document type. Normally they will be called by send operations attempting to fill in holes in the document.

Key Extractor

This utility is used to populate the KeyInstances table. KeyInstances of documents are used for creating abbreviated views of documents and supporting ad hoc queries against them.

5 This utility will take an argument for a document ID. With the document ID it will query the DocTypeKeys table for locations of the key elements. It will then extract the key elements and store them in the KeyInstances table.

Advertising and Content Delivery

10 Systems and processes of the present invention not only track ordering and paying for advertising and other content, or any other product; they also include systems and processes for delivering it. According to a preferred embodiment of the invention for delivering ad copy for the cable television industry, which may be used
15 in concert with the ordering and payment tracking systems mentioned above, systems and process according to the present invention can replace overnight shipment of video tape ads with fast, reliable, all-digital transport of video assets between cable ad sales offices, interconnect offices and head-ends. The digital content can be transmitted to the local head-ends via a hybrid satellite/terrestrial IP
20 data network, and imported directly into digital ad insertion equipment. However, such content distribution systems and processes are more than just a media delivery service. Using an intuitive web Interface, users can securely schedule video delivery, check delivery status, define delivery groups, submit new media content, track distribution progress, and/or archive the content for future use. As shown in

Figure 3, a preferred embodiment of a content distribution system according to the present invention contains four major sub-systems:

- Video Submit Station
- 5 • Multicast Distribution System
- Remote Archive System
- Receive Server

10 The Video Submit Station which may be located at the operations center is used to transmit the media content by way of a high speed terrestrial line to the Network Operations Center (NOC) and schedules delivery of media content to the appropriate local head-ends. The Multicast Distribution System located at the NOC transmits the encoded media content via satellite, the internet or as otherwise desired to the appropriate head-ends per the schedule established by the head-end's Traffic & Billing system. 15 The Remote Archive System, also located at the VNI NOC, stores the encoded media content for future use by TTV personnel. The Receive Server/Router located at the head-end notifies the Multicast Distribution System via the Internet (or other telecommunications pathway), when media content has been received successfully (or not). It also forwards the received data via LAN/WAN to 20 local video servers and caches the locally for later recall.

Features

Distribution Management

Content distribution systems and processes according to the present invention allow control of every aspect of the distribution process. A web-interface
5 allows the user to establish the distribution receive servers, view the archive and schedule delivery. The on-line transaction logs allow the user to track video from submission to VNI to delivery at the head-end.

Data Channel

The systems and processes provide the ability to exchange business critical
10 data between operations and the remote head-ends. Data such as traffic schedules and play-to-air logs are commuted on a daily basis. To better manage the entire process, the head-end's spot inventories can be updated at the NOC continually.

Network Management

The systems and processes can be remotely monitored by on a 7x24
15 schedule. Errors are caught and reported when any of the following events occur:

- Change in network status
- Satellite signal strength falls outside thresholds
- Satellite carrier level falls outside thresholds
- Spot delivery fails
- Schedule and Logs communication fails
- Router or Server processes fail

System Attributes

The functionality and features described in the previous sections can be supported by these system components among others.

Video Submit Station

5 The Video Submit Station is the media management system. The key attributes of the Video Submit Station are described below.

Media Submission

10 All Video Submit tasks can be executed using the system's web Interface or the server-to-server API set called the Video Drop Box. The web interface is an intuitive, easy-to-use screen, with well-organized menus, toolbars, and windows. When commercials are submitted, they are identified with a title ID, unique spot ID number, and optional description. The system provides the user the options to schedule distribution at this time, or store the media for distribution at a later date.

Video Drop Box

15 The Video Drop Box receives spots from the current popular digital insertion systems in cable (Seachange and Skyconnect) or any such systems when submitted and registers each automatically. In this manner, a spot needs only to be encoded and submitted. Then whenever that online spot is referenced in the Insertion
20 schedules it is automatically scheduled for distribution to the headend inserters.

 This system operates invisibly to the insertion operator. The operator encodes spots and selects batches of spots for transfer to the HQ or MVL system (to

name a couple) and the Video Drop Box picks up those submissions and acts automatically to distribute them to the headends.

Encode Formats, Rates and Length

The system can support several formats including MPEG 1, MPEG 1.5 and
5 MPEG 2 as well as JPEG, M-JPEG and any proprietary data formats used in the industry. The bit-rate of the commercials (or other material) is immaterial to the transport system and spots with data rates of 500Kbps through 50 Mbps can be transported. Higher data rates will take longer to transmit in a fixed data rate scenario such as is being proposed here.

Media Storage and Inventory

Using the web interface, the user can specify the length of time the media content is to be archived. The system allows the user to search for keywords in the title and description located within the inventory. The user can also search by various media attributes such as format, bit-rate or date received.

Distribution Setup

All head-ends intended for video distribution are identified as receive servers in the system. The system allows operations to combine many head-ends into groups or create ad-hoc groups dynamically for easy distribution. The user can identify groups by region, client or time period. Once a group is no longer necessary
20 it can be deleted from the list.

Distribution

Once stored media is located in the NOC's inventory it can be scheduled for distribution to the established receive server or groups. This allows for media to be encoded and submitted days prior to its play date at the head-end. This ensures the spot is ready for distribution in the event the media needs to be re-distributed to a given head-end.

Distribution Schedules

Users can access the Distribution Schedule with the web interface. The user can view the spots for distribution on-line with delivery time estimates. Commercials that are scheduled for distribution can be reviewed, edited, re-prioritized or deleted from distribution. Once the media has been delivered, the entry disappears from the schedule.

Delivery Priority

The Multicast Distribution System is based on an algorithm which calculates the distribution priorities based on the T&B schedule for when, the number of head-ends, channels and times the spot will air. The user has the ability to override the computer generated schedule.

Transaction Logs

The system gives the user the ability to track the status of media as it is received by the head-end. Status can be queried based on date, head-end or media ID. This interface allows the user to query the storage requirements of media elements in the archive.

Multicast Distribution System

The Multicast Distribution System according to a preferred embodiment of the present invention is a hybrid satellite/Internet communications system incorporating a reliable transport protocol to ensure the successful delivery of content packages and other information to head-ends. The key attributes of the Multicast Distribution System are described below.

Multicast Routing

The Multicast Distribution System employs the IP multicast technique via satellite to dynamically establish groups of head-ends and transmit media content to them. When submitting media content, the Video Submit user can designate the group(s) of head-ends to which the media is to be delivered.

Digital Satellite Uplink

The Multicast Distribution System employs a digital satellite uplink system for one-way transmission to the head-ends. The uplink transmits a QPSK-modulated waveform in compliance with DVB standards.

Return Channel

There are two primary scenarios for return path data. The first is a standard Internet based return path. This consists of a dial-up connection from each receive site to the nearest ISP for local calling access. This keeps the connection costs to a minimum. The second option for return path connectivity proposed here utilizes VSAT transport.

Internet Model

Certain events will cause the Receive Server to establish a return channel back to the Multicast Distribution System via a local ISP dial-up connection to the Internet. This connection enables the Multicast Distribution System to conduct two-way communications with the Receive Server. This requires either an ISP or a long distance connection.

VSAT Model

There is a negotiated connection from the headend back to the Network Operations Center where the hub would be located. When a need to communicate occurs, the VSAT system negotiates for a chance to communicate back over the satellite channel. This connection enables the Multicast Distribution System to conduct two-way communications with the Receive Server.

Reliable Transport Protocol

The Multicast Distribution System utilizes a reliable transport protocol to facilitate the delivery of media content and other information to the head-ends. This protocol enables the Multicast Distribution System to keep track of which head-ends have received transmissions successfully and which head-ends are having trouble. Upon receipt of a transmission, the Receive Server sends a message back to the Multicast Distribution System via the return channel to confirm that the transmission was successful or to indicate which packets (portions) of the transmission were not received successfully. The Multicast Distribution System re-transmits missed individual data packets as needed until all head-ends have received the transmission successfully, or until a threshold for the number of retries has been reached.

Remote Archive System

The Remote Archive System is located at the NOC and is available to operations for storing media content. The Remote Archive System has the following key attributes:

- Access via a web interface
- Controlled access to archive content based on user profile
- Keyword search capability
- Allocated storage space based on duration defined at time of submission.

Receive Server/Router

There is a Receive Server/Router located at each of the remote head-ends to receive media content. The key attributes of the Receive Server are described below.

Return Channel Connection Manager

When alerted that a transmission is targeted for it, the Receive Server automatically establishes a return path connection to the Multicast Distribution System. In the Internet scenario if the local ISP's connection is unavailable, the Receive Server will automatically dial a pre-configured 800 number for access to the ISP located in the NOC. This helps ensure success of the return channel connection in the Internet scenario, even in the event of a local ISP outage.

The Receive Server/Router also routes data traffic from the satellite data channel to local LAN or WAN based network devices (routers, servers, etc.) using

standard network protocols such as Ethernet and IP. Its on-board hard disk also acts as a cache of received content in cases where immediate routing of data to local devices is unavailable.

Digital Satellite Reception

5 The Receive Server includes a receive-only satellite antenna and digital satellite receive card capable of processing DVB-compliant waveforms. The Receive Server will receive its satellite feed from GE-1.

Another Distribution System

10 Another distribution solution according to the present invention provides hard interconnect functionality to a metro market without the expense and limitations imposed by traditionally implemented hard interconnects. As a bonus, it is far more affordable and can be implemented more quickly than a terrestrial fiber solution. Functionality includes the ability to distribute schedules and spots, gather verification
15 information, handling of agency orders electronically, and simplified spot reconciliation against orders to only list a few benefits.

Fig. 4 shows content being submitted to operations center in two ways. The first (a) is the more traditional method of air courier service. Once at operations, the content is encoded and the Spot ID is assigned, such as according to its ISCI code.
20 The other method (b) shows video submitted using a frame-relay network. With this method Agencies have fixed bit-rate encoders that automatically encode and submit video. Once at operations the spot is transcoded into the proper format and the Spot ID is added.

As shown in Fig. 5, video from the agency (D&S) and order instructions from the media provider's interconnect, are submitted to the Network Operations Center (NOC). Since there are standards discrepancies with today's local MPEG-based insertion systems, systems according to the present invention preferably support any
5 desired platforms for export including MPEG2 Program Streams, MPEG2 Transport Streams and MPEG 1.5. Maintaining customer local site profiles to ensure the correct MPEG file is delivered in the correct local system format is helpful to enabling this feature. The ISCI code will also be embedded in the data stream for later use in verification. Ads will remain in inventory until a buy is associated with the ad (ISCI
10 code).

Once in the proper format, and working in conjunction with the system, a media delivery scheduler will build multicast groups. The system will create these groups based upon daily interconnect buys and deliver the spots over satellite. Delivery verification and status information will be available to interconnect real-time
15 throughout the delivery process. If for any reason video cannot be delivered within a defined set of parameters (i.e. three tries within 3-hours) the system can send the spot on tape via shipper to that site.

The regions identified for placement may be chosen based upon a hybrid ratings tool that compiles market demographic data and ratings information to facilitate
20 optimized market buys. This pre-buy tool can be available on-line. In turn this data can be used by agencies in conjunction with existing schedule building tools from the interconnect.

As shown in Fig. 6, sites can receive their media via satellite. Once received at the ASO or even subtending headend, a utility notifies system personnel that spots have been delivered. The spots can be imported into the MVL or local video archive and matched against the scheduling data received electronically prior to video transmission. If additional subtending sites are present and connected via fiber, customers will conduct business as usual from this point. Otherwise they may wish to have delivery of the spots directly to the subtending sites. If desired, the process of delivering content to subtending sites automatically based upon orders to specific ASO's / regions can be performed.

If at the local level the spot is renamed, this need not effect the verification process as the ISCI code is imbedded in the playback stream. This process is suggested to eliminate problems associated with ASO's or local headends changing the spot title or ID in the T&B system.

As shown in Fig. 7, a "sniffer" box can be placed on the back-end of the insertion playback device and continually search for embedded ISCI codes in the playback stream. When a code is detected, it will record the time, ISCI, length of playback and network. This data will be saved and the logs sent to the ASO, then on to operations. These uploads can be scheduled to happen at regular intervals throughout the week (i.e. once daily at 7:00 PM) and can be done over the Internet using the provided ISP account that is part of all VNI satellite receive station configurations.

Once the ISCI verification data is back at the NOC, operations can reconcile the data against the delivery instructions or contract data stored in the master

platform to immediately determine if make-goods are necessary. This data is also immediately routed to the interconnect for processing where the data will go through another reconciliation against the agency order, then processed for billing.

The following is a list of seven key components of the system software, which may reside in master platform systems and processes mentioned above, for among other reasons to coordinate and make seamless distribution of content with its purchase, sale, and payment.

1. Inventory Management: the system manages a database of video and corresponding traffic data for every piece of interconnect spot video placed onto the interconnect. In placing ads, the system ensures that interconnect management is only aware of its reserved inventory and that all orders are cleared against the same. From the local level, summarized interconnect usage can be made available through a simple web interface.
2. Electronic Orders Using if desired functionality on the master platform, the distribution systems may be capable of receiving and placing electronic orders from agencies and to interconnect in a manner requiring no manual re-entry. All orders are processed and cleared against interconnect inventory. This order information is then forwarded to the local systems. Electronic summary confirmations throughout the process can be provided by the system where required.
3. Schedule Optimization: This feature is a predictive tool that analyses the impact of a buy order, then recommends means to accommodate the buy while providing a snapshot of the buys effects on existing and remaining

buys. Using variables such as ordered dayparts, networks, and geography, spots can be shifted in the schedule to allow others to clear unless prevented by buy parameters. Used by interconnect, if this optimizer cannot clear an order as requested, an electronic change request may be forwarded to the buyer with suggestions of what can be bought that will clear. This way it becomes much easier to manager buys across the interconnect while enabling optimal usage.

4. Schedule Dissemination: This functionality requires advanced capabilities at the NOC, interconnect, and at the local operators. Each day the local operators download orders and modifications for interconnect using the network. Since all transactions are electronic, re-keying of interconnect contracts is not required, and data is automatically transferred into the local system. Interconnect contracts are input into the local systems T&B software at the highest priority thus automatically preempting local spots (assuming it's within the 24-hour buy window). From a simplicity standpoint, interconnect contracts contain no reference to the advertiser or agency and thus local operators do not have to establish or maintain a T&B database. Furthermore, contracts will not contain price information so that the local affiliate is not tempted to preempt interconnect spots due to apparently higher rate local advertisers.
5. Copy Instructions: Copy instructions are downloaded each day using the network, and processed with the incoming interconnect contracts and modifications. These copy instructions, received electronically utilize the

agency-assigned ISCI codes that uniquely identify each commercial.

Physical copy and their pertaining instructions are under interconnect's control.

- 5
6. Verification/Makegoods: Using data gathered by the local "sniffer" box, the system will reconcile the information against the orders placed. This can be done in a number of ways but fundamentally it will first gather (or the local system will submit) the information from the local sites and compile it regionally or by ASO. Next the regional data will be pulled into the NOC, using the master platform functionality if desired, processed, then immediately sent to the interconnect for final reconciliation and billing purposes. These log files will be sent to the interconnect on a daily basis. If in the process of gathering this data the system finds a spot did not run for whatever reason, the system will re-schedule makegoods following pre-determined fulfillment measures. Included in this is notification to the sales associates so they can issue makegood requests to the buyer. With verification taking place daily, in-flight makegoods are very possible yielding much higher fulfillment than typical interconnect spot buys in cable.
- 10
- 15

- 20
7. Now that the interconnect has its summary verification information, the final reconciliation process will begin using the interconnect's order records. Invoices may be created in summary or detailed form as required by the agency. Invoices are returned electronically to the agency via the network.

Conclusion

The foregoing has been provided for purposes of disclosing a preferred embodiment of the present invention. Modifications and changes can be made, and will no doubt happen as technology progresses, to the systems and processes disclosed herein without departing from the scope or spirit of this invention. In short, systems which process and store information and its associated meta-information, such as via a common document model, for controlling work flow, sales, distribution, delivery, and / or payment for media products can fall within the scope and spirit of this invention.